Constructing Snake-in-the-box Codes and Families of such Codes Covering the Hypercube

Proefschrift

Ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. J.T. Fokkema
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op maandag 15 januari 2007 om 10.00 uur

door

Loeky Haryanto

Master of Science in Mathematics, University of Florida,

Master of Arts in Teaching, University of Florida,

Gainesville, Florida-United States

geboren te Solo, Central Java-Indonesia.

Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. A.J. van Zanten

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof.dr. A. J. van Zanten,	Universiteit Maastricht, promotor
Prof. dr. S. M. Dodunekov,	Bulgarian Academy of Sciences, Bulgarije
Prof. dr. V. A. Zinoviev,	Russian Academy of Sciences, Rusland
Prof. dr. F. I. Solov'jeva,	State University Novosibirsk, Rusland
Prof. dr. H. C. A. van Tilborg,	Technische Universiteit Eindhoven
Prof. dr. C. Roos,	Technische Universiteit Delft
Prof. dr. C. Witteveen,	Technische Universiteit Delft
Prof. dr. ir. J. Biemond (reservelid)	Technische Universiteit Delft

Constructing Snake-in-the-box Codes and Families of such Codes Covering the Hypercube
Dissertation at Delft University of Technology

This work has been carried out with financial support from the Royal Netherlands Academy of Arts and Sciences (KNAW) under the framework of the Scientific Programme Indonesia - Netherlands (SPIN).

ISBN: 90-8559-265-8

Keywords: Standard Gray codes, symmetric transition sequence, snake-in-the-box codes (snakes), minimum-weight basis, fixed-position property, ordered basis of linear code, Reed-Muller codes, p-cover of hypercube Q_n by disjoint snakes, invariance translation group.

Cover: Two (vertex-)disjoint snake-in-the-box codes (snakes) cover the hypercube Q_4.

To my wife and children:

Ogi (Oktafien Somalinggi),

Bagas (Bagaskara Galois Somalinggi),

and

Laras (Tiara Monita Larasati).

Delft, the Netherlands,

January 2007

Acknowledgments

This dissertation could not have been written without Prof. dr. A. J. van Zanten who introduced me further into the theory of algebraic coding, in particular into this special topic of ordered codes. My special thanks go to the Royal Netherlands Academy of Arts and Sciences (KNAW) for financially supporting this research, and to the Applied Mathematics Department (DIAM), TUDelft, for its hospitality and for providing me with academic facilities during the last two and half years of my work at TUDelft.

My deep appreciation also goes to Mr. Paul Althuis at CICAT and to his project co-ordinators (Franca Post, Manon Post and Rene Tamboer) for their helpful administrative assistance, despite of the fact that there are so many students and Ph.D candidates to deal with.

I also very much appreciate all my colleagues at TUDelft, especially my roommates Julius and Niels who always entertain me with their music, seemingly from the same song album. Neither will I forget my (former) roommates Berd and Sander, with whom I shared the same office at the 10-th floor before I had to move to the floor of DIAM. With respect to my own research, my special appreciation goes to I Nengah Suparta, the only colleague I knew at the TUDelft who was working in the area of finite ordered codes.

This work is also for the memory of my late father, Lukito, who was so proud being educated in the Dutch educational system adapted in Indonesia. Unfortunately, he passed away during my stay in the Netherlands, while I was the only one of his sons and daughters who was not able to be with him in his last minutes. I am also very grateful to my late father-in-law (Somalinggi) and to my mother-in-law (Mathilda) who always prays for my success, just like my mother (Soehartati) uses to do.

Last but not least, I am very grateful to my wife Oktafien, who keeps supporting me and who convinced me to be determined, and not to be worried by the absence of her and of our children (Bagas, 10, and Laras, 9) during my stay in the Netherlands, though their absence is the only thing that brings about feelings of loneliness.

Table of Contents

1. Introduction

1.1 The Main Objectives

This thesis is about the construction of certain ordered binary codes called *snake-in-the-box codes*, or briefly *snakes*. In general, a snake in a graph is an induced cycle that is a simple cycle with no chords. A chord of a cycle $\mathcal{S}$ is an edge which connects two non-consecutive vertices of $\mathcal{S}$. In this thesis, we consider snakes in the *hypercube* Q_n. The vertices of Q_n are all the 2^n binary n-tuples (also called binary words of length n), and two vertices (i.e. two binary n-tuples) are connected with an edge if and only if they differ in just one position. A snake in Q_n is called a snake-in-the-box code.

More precisely, suppose

$$\mathcal{S} = \mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{L-1}$$

is a list of L words in Q_n. The distance $d(\mathbf{w}_i, \mathbf{w}_j)$ between two words $\mathbf{w}_i$ and $\mathbf{w}_j$ in Q_n is defined to be the Hamming distance of the two words. The list distance $l(\mathbf{w}_i, \mathbf{w}_j)$ between the two words is the minimum number of words in L when going from $\mathbf{w}_i$ until $\mathbf{w}_j$, or more precisely

$$l(\mathbf{w}_i, \mathbf{w}_j) := \min\{|i-j|, L-|i-j|\}.$$

Then $\mathcal{S}$ is a snake-in-the-box code if for every i, j satisfying $0 \le i, j < L$,

$$d(\mathbf{w}_i, \mathbf{w}_{i+1}) = 1 \qquad (*)$$

and

$$l(\mathbf{w}_i, \mathbf{w}_j) = 1 \quad \Rightarrow \quad d(\mathbf{w}_i, \mathbf{w}_j) = 1, \qquad (**)$$

where $\mathbf{w}_L$ is identified with $\mathbf{w}_0$. So, a snake-in-the-box code is a cyclic (closed) list of words of Q_n which satisfies the *nearness condition* (*) and the *separability condition* (**).

Any cycle in Q_n can be specified by a sequence of integers indicating the bit which changes when going from one word to the next. The bits in a binary word of length n are numbered by 0, 1, …, n - 1 from left to right. This sequence is called *transition sequence*. Let

$$T = t_1, t_2, \ldots, t_L$$

be the transition sequence for the snake $\mathcal{S}$ of length (or range) L above. Then the words $\mathbf{w}_{i-1}$ and $\mathbf{w}_i$ only differ in coordinate t_i, $0 < i \le L$. Since $\mathcal{S}$ is cyclic, we have $\mathbf{w}_L = \mathbf{w}_0$.

We emphasize that the adjective ‘cyclic’ for a snake here roughly means that there is essentially no ‘first’ or ‘last’ word in the snake, i.e. the successor of the ‘last’ word is the ‘first’ word. Otherwise, the resulting snake is called an *open snake*. However, some authors (e.g. Casella and Potter [5], Kochut [17]) use the term ‘snake(-in-the-box code)’ for an open, i.e. non-cyclic snake. In terms of its transition sequence (cf. expressions (2.5), (2.8) and (2.17)), a snake is cyclic if and only if every integer in the transition sequence occurs an even number of times.

The snake $\mathcal{S}$ is called *symmetric* if its transition sequence T is of the form

$$T = t_1, t_2, \ldots, t_K, t_1, t_2, \ldots, t_K,$$

which implies that $\mathcal{S}$ is cyclic and $L = 2K$. Kautz in [15] calls such a snake *a natural code*. In this case, the second half of the word list of the snake can be obtained from the first half by translating all words in the first half over the vector $\mathbf{w}_K$, if $\mathbf{w}_0 = \mathbf{0}$.

Many authors have studied the problem of determining upper and lower bounds for the maximal length $s(n)$ of a snake in Q_n [e.g. [1, 4, 5, 8, 17, 26, 30, 31 42], and also how to obtain long snakes (e.g. [24,32])..

A generalization of the notion of snake is a *set of (vertex-)disjoint* snakes. A natural question is to ask for the minimal number $a(n)$ of disjoint snakes of equal length which cover all 2^n vertices of Q_n. Such a set is called a *minimal cover* (by snakes) of Q_n. More generally, we call a set of p disjoint snakes of equal length covering Q_n, a *p-cover* of Q_n.

A problem posed by Erdös, is to decide whether Q_n can be covered with at most l disjoint snakes for some fixed value l, i.e. if there is an integer l such that $a(n) \leq l$, for all $n \geq 2$. Wojciechowski in [32] proved that for $l = 16$, the answer is affirmative, Lukito and van Zanten in [40] showed that such a cover can always be established with *symmetric* snakes for $l \leq 32$.

Constructions of snakes in Q_n are mostly based on techniques of 'extending' snakes existing in Q_m for some $m < n$. The existence of these 'basic snakes' may have been established by any means, e.g. by computer search, or just by accident (cf. e.g. [1]). In this way, the best lower bounds for the length of a snake in Q_n have been derived. Similar techniques are applied in [4], [5] and [16] for the construction of snakes, and more generally, for the construction of circuit codes. Also Wojciechowski's approach for proving the existence of covers of Q_n by snakes is not based on a straightforward construction of concrete snakes.

A different approach for the construction of snakes is presented by Paterson and Tuliani in [24]. This approach exploits the symmetry properties of *necklaces*. A necklace is an ordered list of the words of a constant-weight binary cyclic code. Here also a computer search is used to get some basis objects. In Chapter 3, we shall generalize the notion of a necklace and also slightly generalize the method of [24]. Moreover, we shall prove that the two constructions in [24] that are used to produce snakes are equivalent.

Our major goal in this thesis is to construct snakes in a more straightforward way, i.e. by a non-recursive method. Moreover, we require our method to be extendable for the construction of covers of Q_n by vertex-disjoint snakes, possibly improving the result of Wojciechowski in [32].

Our construction starts from a minimum-weight-d basis of some linear algebraic code $\mathcal{C}$. The weight-d basis vectors are arranged according to a standard Gray code, resulting in an

ordered cyclic list of the codewords of $\mathcal{C}$, such that each codeword is at Hamming distance d from the previous one. This framework constitutes the skeleton of the snake to be constructed.

In order to obtain the whole snake, we have to change the d bits, going from one codeword of $\mathcal{C}$ to the next one, in such an order that the separation property or separability condition (no chords) (**) of a snake is satisfied. Various variations and generalizations of this method seem to be possible.

In this thesis, we apply the method above for $d = 4$. Apart from being symmetric due to the applied standard Gray code, our snakes have some additional structure, because of the linearity of the underlying code $\mathcal{C}$. To obtain disjoint snakes and to construct covers of Q_n with symmetric snakes, this structure can be exploited occasionally. Such covers will be called *symmetric covers*.

1.2 Outline of the Next Sections

The contents of this thesis is distributed among the various sections in the following way. Since our construction methods heavily rely on *Gray codes*, we discuss in Section 2.1 the well known *standard or binary–reflected* Gray code $G(n)$, which is a major tool throughout the thesis.

In Section 2.2 the so-called 'index problem' of $G(n)$ is discussed, i.e. the relationship between a codeword of $G(n)$ and its index in the (cyclic) list of codewords. This relationship is also well-known. In Section 2.3 we derive and prove a number of simple properties of $G(n)$ which to the best of our knowledge are not referred to in the literature, while in Section

2.4 various special cases of Gray codes are introduced and briefly discussed. One of these codes is the snake-in-the-box code as defined in Section 1.1.

The method of Paterson and Tuliani mentioned in Section 1.1, is extensively discussed in Section 3.1, and generalized in Sections 3.2 and 3.3.

In Section 4.1, the details of our method are discussed to construct snakes based on a linear $[n, k, d]$-code, while in Section 4.2 a number of conditions are developed that are necessary and/or sufficient to apply this method in the case $d = 4$. The index problem for the resulting snake-in-the-box codes is solved in Section 4.3, where we used the solution of the index problem for $G(n)$ as presented in Section 2.2.

Concrete examples of snakes in the hypercubes Q_n, $3 < n \leq 16$, are produced in Chapter 5. It turned out that the *Reed-Muller codes* $R(m - 2, m)$, with $2^m = n$, are appropriate linear codes to start with when applying our construction. Therefore we briefly discuss Reed-Muller codes in Sections 5.1 and 5.2, as well as the existence of what we call *parallel systems* in *Euclidean Geometries* in Section 5.3. These parallel systems are relevant for choosing a special basis of $R(m - 2, m)$ satisfiying the so-called *fixed-position property*. In Section 5.4, snakes are constructed for the hypercubes Q_5, Q_6, Q_7 and Q_8, and in Section 5.5 we do the

same for the hypercubes, Q_9, Q_{10}, Q_{11}, …, Q_{16}.

In order to construct covers for Q_n consisting of disjoint snakes, we develop in Sections 6.1, 6.2 and 6.3 a number of criteria to decide when a snake and a translated snake (which has the same transition sequence) are disjoint. In Section 7 we apply these criteria for the construction of 4-covers for Q_n, $5 \leq n \leq 8$, and of 8-covers for $9 \leq n \leq 15$. in 7.1. In the case of Q_{16}, we first construct in Section 7.2 a number of *near-covers* consisting of 8 snakes not all of which are mutually disjoint. By taking a slightly different basis for the underlying [16, 11. 4]-code, we succeeded in constructing a real 8-cover for Q_{16}. This symmetric 8-cover is presented in Section 7.3.

Finally, in Section 8, we use the knowledge built-up in Sections 4 - 7 to describe a straightforward general construction of snakes in Q_n and of covers of Q_n by snakes for $n > 3$. In Section 8.1 we discuss the role of the Griesmer bound in our approach. In Section 8.2 we show that a snake-in-the-box code and its translates which cover a hypercube Q_n can be connected to each other in such a way that the result is a complete cyclic Gray code in Q_n. From this point of view, a cover of Q_n by snakes originates in quite a natural way from a special Gray code. This idea is exploited in Section 8.3 to come to a general theorem about covers of Q_n and the accompanying invariance group.

It turns out that for $4 < n \leq 8$ and for $8 < n \leq 16$, the results are better than in [32], i.e. the number of snakes in the cover of Q_n is equal to 4 and to 8, respectively, whereas in [32], it is only stated that this number is upperbounded by 16. Even in the range $16 \leq n \leq 32$, one could say that Corollary 8.8 gives slightly better results, since the 16 covering snakes are symmetric, and so we have a symmetric 16-cover. Moreover in the range, $4 \leq n \leq 16$, Corollary 8.8 gives a supplement to a result in [2] which states that for any even integer $r \geq 4$, $n \geq 2$, the graph K_r^n, i.e. the n^{th} power of the complete graph K_r, can be covered with r^3 (not necessary pairwise vertex-disjoint) snakes. The results obtained in Chapters 7 and 8 have been collected in [38] (cf. also [37]).

Although this thesis is not about constructing “long snakes”, or improving the known upper and lower bounds for the maximal snake length $s(n)$ in Q_n for certain values of n, we collected the main results concerning $s(n)$ in Appendix A (cf. also [4], [5], [8], [19], [30], [31] and [42]). The reason for this is that occasionally knowledge about $s(n)$ can help deciding whether a certain cover of Q_n is optimal, i.e. whether the number of snakes in that cover is minimal.

As for our notation, we shall stick to the widely used convention to label the bits of a codeword of the standard Gray code $G(n)$ by 1 until n, from *right to left*, since this is very convenient for all properties which are proven by induction or recursion. On the other hand, if we are dealing with snake-in-the-box codes in Sections 4 until 8, we shall label the bits of the

codewords by 0 until $n - 1$ from *left to right*, since these codewords originate from a vector space $GF(2)^n$ where the labeling of vector components is similar. Since the standard Gray code $G(k)$ is only used as an auxiliary tool to label blocks of snake words, the two different conventions do not interfere.

A somewhat unlucky exception is the labeling of bits in snake words in Chapter 3. Here, the work of Paterson and Tuliani is discussed and generalized, and therefore we stick to their conventions which implies that the labeling of the bits runs from 1 until n, occasionally from left to right.

2. Gray Codes and Circuit Codes

In this chapter, we discuss briefly the definition and a number of well-known properties of the standard or binary reflected Gray code in Sections 2.1 and 2.2. In Section 2.3, we prove some properties of $G(n)$ which we did not find in the literature Section 2.4 gives a review of special Gray codes satisfying additional requirements.

2.1 The Standard Gray Code $G(n)$

Let Q_n be the n-dimensional cube (*hypercube*), or shortly n-cube. This is the graph with all 2^n binary words of length n, or n-tuples, as vertices and all pairs of vertices which differ exactly in one coordinate as edges. These n-bit numbers are called (*binary*) *codewords*. An ordered binary code of *range* L is a sequence or list of L codewords

$$\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{L-1}.$$

A Gray code can be defined as a Hamilton cycle in Q_n, i.e. circuit in Q_n containing the 2^n vertices precisely once. Equivalently, a *Gray code* $G(n)$ is a sequence of n-bit words such that two successive words differ in precisely one position.

The best known example of such a code is the binary *reflected*, or the *standard Gray code*. If we define

$$G(1) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{2.1}$$

then the code $G(n)$, $n > 1$, can be recursively defined as the list of ordered rows of an $2^n \times n$ matrix

$$G(n) = \begin{pmatrix} 0 & G(n-1) \\ 1 & G^R(n-1) \end{pmatrix}, \tag{2.2}$$

where $G^R(n-1)$ stands for the reversed list of $G(n-1)$, e.g. the i-th word of $G^R(n-1)$ is the $(2^{n-1} - 1 - i)$-th word of $G(n-1)$, $0 \le i \le 2^{n-1} - 1$.

Given $n \ge 1$ and a fixed n-bit word $\mathbf{g}_0$, all 2^n codewords of $G(n)$ can also be generated by the *symmetric* (or *non-cyclic*) transition sequence S_n as follows. This non-cyclic transition sequence of $G(n)$ can be defined recursively by firstly defining

$$S_1 := 1 \tag{2.3}$$

and for every positive integer $n > 1$,

$$S_n := S_{n-1}, n, S_{n-1}, \tag{2.4}$$

If we write

$$S_n := t_1, t_2, \ldots, t_{2^n-1}, \tag{2.5}$$

then given $\mathbf{g}_0 = \mathbf{0} = 00\ldots0$, the words in the standard Gray code

$$G(n)=\mathbf{g}_0,\mathbf{g}_1,\ldots,\mathbf{g}_{2^n-1} \tag{2.6}$$

are consecutively constructed such that for every $x = 1, 2, \ldots, 2^n - 1$, the nonzero word $\mathbf{g}_x$ is determined by the integer t_x in (2.5) after changing the t_x-th bit of $\mathbf{g}_{x-1}$.

A very useful form of S_n is obtained by splitting it into 2^{n-i} subsequences S_{i-1} in the following way

$$S_n = S_i, i+1, S_i, i+2, \ldots, S_i, n, S_i, \ldots, i+1, S_i. \tag{2.7}$$

This form can be derived by consecutively applying (2.4) to the subsequences $S_{n-1}, S_{n-2}, \ldots, S_i$ in the RHS of the expression (2.7).

The standard Gray code $G(n)$ has the property that its last word differs from the first word in precisely one bit, i.e. the n-th bit (we label the bits or the positions in a codeword of Gray codes from 1 until n, from right to left). So, we can interpret $G(n)$ as a *cyclic* Gray code, with *complete* transition sequence

$$\overline{S}_n := S_n, n. \tag{2.8}$$

2.2 Index Problem of $G(n)$

The 2^n codewords of $G(n)$ are labeled by an index which runs from 0 until 2^{n-1}, as indicated in (2.6). A natural problem is to determine the codeword $\mathbf{g}_x \in G(n)$ when the index value x is given. We call this problem together with the reverse problem of determining the index x of a given word $\mathbf{g}_x$ in the list $G(n)$, *the index problem of* $G(n)$.

The solution of the problem is well-known and can be found in some textbooks, e.g. in [18, 27]. We state and prove this result in the following way, where we shall write, for reasons of convenience, g_i instead of $(\mathbf{g}_x)_i$, $1 \le i \le n$, for the components of $\mathbf{g}_x$.

Theorem 2.1

Let x be some integer with $0 \le x < 2^n$, $n \ge 1$, and let

$$\mathbf{g}_x = g_n g_{n-1} \ldots g_1$$

be the corresponding codeword in the standard Gray code $G(n)$. If

$$\mathbf{x} = x_n x_{n-1} \ldots x_1$$

is the binary representation of the integer x, then

$$g_i = x_i + x_{i+1} \pmod 2, \tag{2.9}$$

for all i with $1 \le i \le n$, where $x_{n+1} := 0$.

We first remark that the rule to compute $\mathbf{g}_x$ as stated in Theorem 2.1, can also symbolically be formulated as $\mathbf{g}_x = \mathbf{x} \oplus \lfloor \mathbf{x/2} \rfloor$, where $\mathbf{x}$ stands for the binary representation of x and $\lfloor \mathbf{x/2} \rfloor$ stands for the binary representation of $x/2$, while '$\oplus$' denotes the bitwise addition modulo 2 (exclusive-or operator) of two vectors. Also observe that $g_n = x_n$.

Proof. We shall prove the rule in this theorem by induction. Now assume that the Theorem is true for $G(n)$, $n > 1$. If

$$G(n+1) = \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{2^{n+1}-1}$$

is expressed as a matrix of the form (2.2), then from the definition of

$$G^R(n) = \mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{2^n-1},$$

the 'mirror image' of $G(n)$, we can choose a pair of words $\mathbf{v}_{x'} = 0\mathbf{g}_x$ and $\mathbf{v}_{y'} = 1\mathbf{h}_y$ of $G(n+1)$ such that

$$\mathbf{g}_x = \mathbf{h}_y = g_n\, g_{n-1} \ldots g_1,$$

where $0 \le x, y \le 2^n - 1$, $0 \le x' \le 2^n - 1 < y' \le 2^{n+1} - 1$. Here, x and y are related via the equality

$$y = 2^n - 1 - x$$

whereas x' and y' are related via the equality $x' = x$ and

$$y' = 2^n - 1 + y = 2^{n+1} - 1 - x.$$

From the fact that $2^n - 1$ is represented by the binary n-tuple 11…1, it follows that if

$$\mathbf{x} = x_n\, x_{n-1} \ldots x_1$$

and

$$\mathbf{y} = y_n\, y_{n-1} \ldots y_1$$

are the binary representations of x and its corresponding integer y, respectively, then

$$\mathbf{y} = (1 - x_n)\,(1 - x_{n-1}) \ldots (1 - x_{n-1}). \tag{2.10}$$

Since $0 \le x' \le 2^n - 1 < y' \le 2^{n+1} - 1$, it follows that if

$$\mathbf{x}' = x'_{n+1}\, x'_n \ldots x'_1 = x'_{n+1}\, x_n \ldots x_1$$

and

$$\mathbf{y}' = y'_{n+1}\, y'_n \ldots y'_1 = (1 - x'_{n+1})(1 - x_n) \ldots (1 - x_1)$$

are the binary representations of x' and y', respectively, we have $x'_{n+1} = 0$ and $y'_{n+1} = 1$. This means that $y'_{n+1} = 1 = 1 - x'_{n+1}$. On the other hand, expression (2.10) states that for every i with $0 \le i \le n$, we have $x_i = (1 - y_i)$. We conclude that for every i with $0 \le i \le n + 1$,

$$x_i = x'_i = (1 - y'_i) = (1 - y_i).$$

By the induction assumption, the equality of the form (2.9) relating $\mathbf{v}_{x'} = 0g_n\, g_{n-1} \ldots g_1$ with the n-tuple $\mathbf{x}'$, and also relating $\mathbf{v}_{y'} = 1g_n\, g_{n-1} \ldots g_1$ with the n-tuple $\mathbf{y}'$, is true for all components $v_i = g_i$ with $0 \le i \le n$, e.g. the n-th bit g_n of $\mathbf{v}_{y'} = 1g_n\, g_{n-1} \ldots g_1$ is the sum of the n-th and the $(n+1)$-bits of $\mathbf{y}'$,

$$g_n = (1 - x'_{n+1}) + (1 - x_n) = x_n + x'_{n+1} = x_n.$$

It remains to show that (2.9) is also true for the $(n+1)$-th bits of $\mathbf{v}_{x'}$ and $\mathbf{v}_{y'}$. But given $x'_{n+2} = 0$, since $x'_{n+1} = 0$, the $(n+1)$-bit 0 of $\mathbf{v}_{x'}$ satisfies the equality

$$0 = x'_{n+1} + x'_{n+2}.$$

So, the expression (2.9) is true for the first half of $G(n+1)$. Also given $y'_{n+2} = 0$, since $y'_{n+1} = 1$, the $(n+1)$-bit 1 of $\mathbf{v}_{y'}$ satisfies the equality

$$1 = y'_{n+1} + y'_{n+2}$$

i.e. the expression (2.9) is true for the second half of $G(n + 1)$. Since (2.9) is trivially true for $n = 1$, we are done. □

We remark that the index-word conversion (2.9) in Theorem 2.1 can also be represented by the equality

$$\mathbf{xM} = \mathbf{g}$$

where $\mathbf{g}$ and $\mathbf{x}$ are considered as (row) vectors of $GF(2)^n$ and where

$$\mathbf{M} = \begin{pmatrix} 1 & 1 & 0 & \vdots & 0 \\ 0 & 1 & 1 & \vdots & 0 \\ 0 & 0 & 1 & \vdots & 0 \\ \cdots & \cdots & \cdots & \vdots & \cdots \\ 0 & 0 & 0 & \vdots & 1 \\ 0 & 0 & 0 & \vdots & 1 \end{pmatrix},$$

is an $n \times n$ matrix over $GF(2)$. Since it is clear that the inverse of $\mathbf{M}$ is equal to

$$\mathbf{M}^{-1} = \begin{pmatrix} 1 & 1 & 1 & \vdots & 1 \\ 0 & 1 & 1 & \vdots & 1 \\ 0 & 0 & 1 & \vdots & 1 \\ \cdots & \cdots & \cdots & \vdots & \cdots \\ 0 & 0 & 0 & \vdots & 1 \\ 0 & 0 & 0 & \vdots & 1 \end{pmatrix},$$

the solution for the index-word conversion immediately follows.

Theorem 2.2

Let $\mathbf{g}_x$ be some codeword of the standard Gray code $G(n)$, $n \geq 1$. Then the binary representation of its index x is determined by

$$x_i = g_i + g_{i+1} + \ldots + g_n \pmod 2,$$

for all i, $1 \leq i \leq n$.

Example 2.1

In order to determine $\mathbf{g}_{183} \in G(8)$, we first need the binary representation

$$\mathbf{x} = x_8\, x_7\, x_6\, x_5\, x_4\, x_3\, x_2\, x_1$$

of $x = 183$. Since $183 = 2^7 + 2^5 + 2^4 + 2^2 + 2^1 + 2^0$, we conclude that

$$\mathbf{x} = 10110111$$

from which we have $x_1 = 1$, $x_2 = 1$, $x_3 = 1$, $x_4 = 0$, $x_5 = 1$, $x_6 = 1$, $x_7 = 0$, $x_8 = 1$. Theorem 2.1 yields that

$$\mathbf{g}_{183} = \mathbf{x} \oplus \lfloor \mathbf{x}/2 \rfloor = 10110111 + 01011011 = 11101100.$$

Conversely, suppose

$$\mathbf{g}_y = 11011101.$$

If $\mathbf{g}_y = g_1\, g_2 \ldots\, g_8$, then

$$g_1 = 1, g_2 = 0, g_3 = 1, g_4 = 1, g_5 = 1, g_6 = 0, g_7 = 1, g_8 = 1.$$

If $\mathbf{y} = y_8\, y_7\, y_6\, y_5\, y_4\, y_3\, y_2\, y_1$ is the binary representation of y, then using the formula in Theorem 2.2, we find

$$y_8 = g_8 = 1 \bmod 2 = 1,$$
$$y_7 = (g_7 + g_8) \bmod 2 = (1 + 1) \bmod 2 = 0,$$
$$\ldots,$$
$$y_1 = (g_1 + g_2 + g_3 + g_4 + g_5 + g_6 + g_7 + g_8) \bmod 2 = 0.$$

Therefore,

$$\mathbf{y} = 10010110,$$

is the binary representation of $y = 2^7 + 2^4 + 2^2 + 2^1 = 149$.

2.3 Properties of the Standard Gray Code $G(n)$

Let

$$C = \mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{L-1} \tag{2.11}$$

be some, not necessarily complete, cyclic Gray code, i.e. all these words are different and each word differs from its successor in precisely one bit, and moreover, we identify $\mathbf{w}_L$ and $\mathbf{w}_0$, i.e. $\mathbf{w}_0$ is considered to be the successor of $\mathbf{w}_{L-1}$ (cf. also the end of Section 2.1). This last statement implies that L is even.

We define the *list distance* $l(\mathbf{w}_i,\mathbf{w}_j)$ between two words $\mathbf{w}_i$ and $\mathbf{w}_j$ as

$$l(\mathbf{w}_i, \mathbf{w}_i) := \min\{|i - j|, L - |i - j|\}. \tag{2.12}$$

In addition, we also define the well-known Hamming distance

$$d(\mathbf{w}_i,\mathbf{w}_j)$$

between $\mathbf{w}_i$ and $\mathbf{w}_j$, being the number of bits where the two words differ. For example, we have $d(00111, 01011) = 2$ because the second and third bits of 00111 and 01011 are different.

Since the Gray code $G(n)$ is considered as a cyclic list, its 2^n words can also be obtained using the cyclic (or complete) transition sequence (2.8) which can be written as

$$\overline{S}_n := S_{n-1}, n, S_{n-1}, n. \tag{2.13}$$

We notice that we can take any word of length n as initial word $\mathbf{g}_0$. The resulting Gray code is equivalent to the standard Gray code $G(n)$ as defined by (2.3) and (2.4), in the sense that it can be obtained by adding $\mathbf{g}_0$ to all words of the list of $G(n)$.

Definition 2.1

For any sequence T of integers from the following finite set, which is the discrete interval

$$[1, n] = \{1, 2, \ldots, n\},$$

the contents $c(T)$ of T is the set of those integers occurring an odd number of times in T. A similar definition of the contents $c(T)$ applies to $T \subseteq [0, n-1] = \{0, 1, \ldots, n-1\}$.

For example, if $\mathbf{g}_x \in G(n)$ is generated by a subsequence T of $\overline{S}_n$ and $X = sup\ \mathbf{g}_x$, then $X = c(T)$. Here, $sup\ \mathbf{g}_x$ is the support of $\mathbf{g}_x$, i.e. the set of positions where $\mathbf{g}_x$ has components equal to 1. We also adopt the convention that if $T = t_x, t_{x+1}, \ldots, t_y$, then its reversed sequence is

$$T^R = t_y, \ldots, t_{x+1}, t_x.$$

Moreover, if $T \neq \overline{S}_n$ is a non-empty sequence, we define the *complement* of T as the subsequence

$$T^c = t_{y+1}, \ldots, t_{x-2}, t_{x-1}.$$

The notion of complement is valid only for a cyclic transition sequence.

In the next, we shall always consider $\overline{S}_n$ as a cyclic sequence, likewise we also consider the list of codewords $G(n)$ as a cyclic list.

Theorem 2.3

Let $\overline{S}_n$ be the transition sequence of the standard Gray code $G(n)$. For every integer $n \geq 3$, the transition sequence $\overline{S}_n$ can be obtained from $\overline{S}_{n-1}$ in the following ways:

(*i*). *by replacing the two integers $n-1$ in $\overline{S}_{n-1}$ by the subsequence*

$$n-1, S_{n-2}, n$$

or alternatively by the subsequence

$$n, S_{n-2}, n-1;$$

(*ii*). *by replacing all integers* 1 *in* $\overline{S}_{n-1}$ by S_2 *and augmenting all other integers in* $\overline{S}_{n-1}$ *by* 1;.

(*iii*). *by replacing one of the two integers $n-1$ in* $\overline{S}_{n-1}$ by

$$n, S_{n-1}, n.$$

Theorem 2.4

Suppose $\overline{S}_n$ is the transition sequence of the standard Gray code $G(n)$. Then the transition sequence $\overline{S}_n$ satisfies the following properties:

(i) *for any subsequence T of $\overline{S}_n$ that has length 2^{n-1}, we have $\overline{S}_n = T, T$;*

(ii) *$\overline{S}_n$ is invariant for interchanging $n-1$ and n;*

(iii) $S_n = S_n^R$.

The proofs follow immediately from (2.8) and (2.13).

Remark

The equality $\overline{S}_n = T, T$ of Theorem 2.4(*i*) implies that we do not distinguish between $\overline{S}_n$ and any of its cyclic translations. We also remark that Theorem 2.3(*ii*) can equivalently be

formulated by saying that we augment all integers of $\overline{S}_{n-1}$ by 1, and next insert a 1 between any two integers and also put a 1 at the front and at the back.

Theorem 2.5

Let $T := i, T', j$ be a subsequence of the transition sequence $\overline{S}_n$ of the standard Gray code $G(n)$, $n > 1$.

(i) *if $1 < i < j$ or $1 < j < i$, then $|c(T)|$ is odd and min $c(T) = i - 1$ or min $c(T) = j - 1$, respectively;*

(ii) *if $i = j$, then $|c(T)|$ is odd and min $c(T) \geq i$ for $i < n$, whereas $c(T) = \{i - 1\}$ for $i = n$;*

(iii) *if $i = j$, then $c(T) \neq \{i\}$;*

(iv) *if $i = 1$, $j > 1$ or $i > 1$, $j = 1$, then $|c(T)|$ is even;*

(v) *if $T^R = T$, then either $i = j < n$ and $c(T) = \{m\}$ for some $m > i$, or $i = j = n$ and $c(T) = \{n - 1\}$.*

Proof. For $n = 2$, all statements are either void (e.g. (*i*)) or they appear true by inspection. For $n \geq 3$, we proceed by induction. Assume that all statements are true for a certain value $n - 1$ with $n > 3$. First we prove that (*i*) also holds for n. If T does not contain the integer n, this follows immediately from the induction assumption. If T contains the integer n twice, it follows also from the induction assumption applied to the complement T^c of T, omitting the integer 1 at the front and at the back of T^c, and using $c(T^c) = c(T)$. If T contains n once, we can write $T = i, \ldots, n, \ldots, i, 1, \ldots, j$ (or $T = i, \ldots, 1, j, \ldots, n, \ldots, j$) such that $c(i, \ldots, n, \ldots, i) = \{n\}$ (or $c(j, \ldots, n, \ldots, j) = \{n\}$). From the induction assumption, we have that $|c(1, \ldots, j)|$ and $|c(i, \ldots, 1)|$ are even, and hence $|c(T)|$ is odd.

In a similar way, the relations (*ii*) – (*v*) can be proved to hold for n. Since one can easily verify that relations (*i*) – (*v*) hold for $n = 3$. We have proved the Theorem by the principle of mathematical induction. □

For the next theorem, we partition the list $G(n)$ into 2^{n-i} sublists of size 2^i, starting from the zeroword, i.e.

$$G(n) = G_i^0(n), G_i^1(n), \ldots, G_i^{2^{n-i}-1}(n), \tag{2.14}$$

for some i, $1 \leq i \leq n - 1$. The non-cyclic lists $G_i^j(n)$ all have the same non-complete transition sequence S_i of (2.7), and so such a sublist can be obtained from any other one by adding a fixed binary vector from $GF(2)^n$ to all words of the latter list. For reasons of convenience, we introduce the vectors $\mathbf{e}_{k,l}$ in $GF(2)^n$ which have, by definition, ones at positions k and l from the right and zeros elsewhere, where $k, l \in [1, n]$, with $k \neq l$.

Theorem 2.6

The sublists $G_i^j(n)$ *of the standard Gray code* $G(n)$, $n \geq 2$, *are related to each other by the recurrence relation:*

$$G_i^j(n) = G_i^{j-1}(n) + \mathbf{e}_{i,i+t_j}, \qquad 0 < j \leq 2^{n-i}, \tag{2.15}$$

where the integers t_j *are the elements of the transition sequence* $\overline{S}_{n-i}$ *of* $G(n-i)$, *and where* $G_i^{2^{n-i}}(n)$ *is identified with* $G_i^0(n)$.

Proof. For $n = 2$, the Theorem is trivially true. Assume that the recurrence relation holds for some $n \geq 2$, and consider the partitioned list

$$G(n+1) = G_i^0(n+1),\ G_i^1(n+1),\ \ldots, G_i^{2^{n+1-i}-1}(n+1). \tag{2.16}$$

For $i = n$ we have $\overline{S}_{n+1-i} = \overline{S}_1 = 1, 1$, and $\mathbf{e}_{i,i+t_1} = \mathbf{e}_{i,i+t_2} = \mathbf{e}_{n,n+1}$. Indeed, the relation

$$G_n^1(n+1) = G_n^0(n+1) + \mathbf{e}_{n,n+1}$$

is true, as follows immediately from the definition of $G(n+1)$. Next, we take $i = 1$. In this case we have $G(n+1) = G_1^0(n+1),\ G_1^1(n+1),\ \ldots, G_1^{2^n-1}(n+1)$, where each sublist $G_1^j(n+1)$ contains two words differring at position 1. From Theorem 2.3(*iii*), it follows that $\overline{S}_{n+1} = 1, t_1 + 1, 1, t_2 + 1, \ldots, 1, t_{2^n} + 1, 1, t_1 + 1, 1, t_2 + 1, \ldots, 1, t_{2^n} + 1$, where $t_1, t_2, \ldots, t_{2^n}$ are the integers of $\overline{S}_n$ as defined by (2.5) and (2.8). This proves the recurrence relation for $i = 1$.

Finally, for $1 < i < n$, the relation follows easily for the sublists $G_i^j(n+1)$, $0 < j \leq 2^{n-i} - 1$, in the first half of $G(n+1)$ by putting a zero in front of vectors $\mathbf{e}_{i,i+t_j} \in GF(2)^n$. The same holds for the sublists $G_i^j(n+1)$, $2^{n-i} < j \leq 2^{n+1-i} - 1$ in the second half of $G(n+1)$. If $j = 2^{n-i}$, we put a one in front of the vector $\mathbf{e}_{i,i+t_j} \in GF(2)^n$ and remove the one from position $i + t_j$. In this way, we get the translation vector $\mathbf{e}_{i,n+1} \in GF(2)^{n+1}$. Therefore the Theorem also holds for $n + 1$. □

We shall derive next some results which are in a way the reverses of the statements in Theorem 2.5. Notice that from Theorem 2.5, it follows from $X = c(i, T', j)$, $|X| > 1$ and odd, both integers i and j must be distinct and greater than 1.

Theorem 2.7

Let $\overline{S}_n$ *be the transition sequence of the standard Gray code* $G(n)$ *and let* X *be an arbitrary fixed subset of* $[1, n]$, $n \geq 3$, *with* $|X|$ *odd, and* $i_0 := \min X$. *If one defines* $i := i_0 + 1$, *then*

(i) if $1 < i < n$, *then for any* j *with* $i < j \leq n$, *there exists a subsequence* $T = i, T', j$ *of* $\overline{S}_n$ *such that* $c(T) = X$;

(ii) if $1 < i \leq n$, *then for any j with* $i < j \leq n$, *there exists a subsequence* $T = j, T', i$ *of* $\overline{S}_n$ *such that* $c(T) = X$;

(iii) if $1 < i \leq n$, *then for any j with* $1 \leq j \leq i - 1$, *there exists a subsequence* $T = j, T', j$ *of* $\overline{S}_n$ *such that* $c(T) = X$;

Proof.

(*i*) We shall apply induction to $n \geq 3$. For $n = 3$, the statement is true as one can easily verify by inspection. Assume the statement is true for all values less than some $n > 3$. In order to prove the statement for n, we distinguish between the cases $i > 2$ and $i = 2$. Suppose $n > 3$ and for every positive integer $m < n$, we assume the Theorem is true in $\overline{S}_m$.

If $i > 2$, we omit from $\overline{S}_n$ all integers 1. Furthermore, we replace all remaining integers t in $\overline{S}_n$ by $t - 1$, yielding $\overline{S}_{n-1}$ (cf. Th.2.3(*ii*)). We do the same for all integers in X, yielding a set Y with $min\ Y = i - 2$. By the above assumption, there is a subsequence $S = i - 1, S', j - 1$ of $\overline{S}_{n-1}$ such that $c(S) = Y$. Now, we apply the reverse of Th. 2.3(*i*) to S, yielding a subsequence T of $\overline{S}_n$ which satisfies all requirements. (Notice that the number of integers 1 inserted in S is always even). So, in the case $i > 2$ statement (*i*) has been proved for n.

For $i = 2$, we distinguish four subcases.

1. $j < n,\ n \notin X$; 2. $j < n,\ n \in X$; 3. $j = n, n \notin X$; 4. $j = n, n \in X$.

1. The case $j < n,\ n \notin X$.

   We can apply th induction assumption to one of the two subsequences S_{n-1} of $\overline{S}_n$.

2. $j < n,\ n \in X$.

   2. 1. $j = n - 1, n \in X$.

   Define $Y = \{n - 1\} \cup X \setminus \{n\}$ in case $n - 1 \notin X$, and $Y = X \setminus \{n - 1, n\}$ in case $n - 1 \in X$. In both cases, we have $|Y|$ is odd, and hence we can apply the induction assumption. It follows that in the first case, there is a subsequence $S = 2, S', n - 1$ of $\overline{S}_{n-1}$ such that $c(S) = Y$. Since $n - 1 \in Y$, we have that S' does not contain $n - 1$. So, the sequence

$$T = 2, S', n - 1, S_{n-2}, n, S_{n-2}, n - 1$$

   is a subsequence of $\overline{S}_n$. It will be clear that $c(T) = \{n\} \cup Y \setminus \{n - 1\} = X$, and hence T satisfies all requirements.

   In the second case, there also exists, by the inductive assumption, a subsequence

$$S = 2, S', n - 1$$

of $\overline{S}_{n-1}$ such that $c(S) = Y$. Since now $n-1 \notin Y$, we have that $n-1 \in S'$, and so the sequence

$$T = 2, S', n$$

is a subsequence of $\overline{S}_n$ with $c(T) = Y \cup \{n-1, n\} = X$. Here, we applied Theorem 2.3(*iii*). Hence, the subsequence satisfies all requirements.

2. 2. $j < n-1, n \in X$.

Like in Case 2.1, we define $Y = \{n-1\} \cup X \setminus \{n\}$ in case $n-1 \notin X$ and $Y = X \setminus \{n-1, n\}$ in case $n-1 \in X$. In both cases, we have again that we can apply induction, since $|Y|$ is odd. In the first case, there exists by induction a subsequence $S = 2, S', j$, which is a subsequence of $\overline{S}_{n-1}$, such that $c(S) = Y$. Since $j < n-1$, this implies that $n-1 \in S'$. Replacing this integer $n-1$ (and also the second integer $n-1$) by n results in a subsequence T of S_n, by Theorem 2.4(*ii*) and eq. (2.13). This subsequence T has the property $c(T) = \{n\} \cup Y \setminus \{n-1\} = X$, and hence T satisfies all requirements.

In the second case, we argue as follows. By induction, there exists a sequence

$$S = 2, S', j,$$

which is a subsequence of S_{n-2} such that $c(S) = Y$. We write $\overline{S}_{n-2} = U, S, V$ and we define a sequence

$$T = S, V, n-1, U, S, V, n, U, S,$$

which is a subsequence of $\overline{S}_n$ according to eqs. (2.8) and (2.13). It is obvious that $c(T) = c(S) \cup \{n, n-1\} = Y \cup \{n, n-1\} = X$. Since the first integer of T is 2 and the last integer is j, the sequence T satisfies all requirements.

3. $j = n, n \notin X$.

Since S_n generates all binary words of length n, there exists in S_n a subsequence

$$S = 1, 2, 1, \ldots, l, 1$$

for some $l > 2$, starting with the first integer 1 of $\overline{S}_n$, such that $c(S) = X$. The complement T of S w.r.t. $\overline{S}_n$ has the form

$$T = 2, 1, \ldots, l, \ldots, 1, n, 1, 2, 1, \ldots, 1, 2, 1, n$$

also satisfies $c(T) = X$, and so this sequence satisfies all requirements.

4. $j = n, n \in X$.

Like in Case 2.1 and Case 2.2, we define $Y = \{n-1\} \cup X \setminus \{n\}$ in case $n-1 \notin X$ and $Y = X \setminus \{n-1, n\}$ in case $n-1 \in X$. In the first case, there exists, by induction, a subsequence $S = 2, S', n-1$ of $\overline{S}_{n-1}$ such that $c(S) = Y$. It is obvious that $n-1$ does not occur in S'. Replacing $n-1$ by n in S yields a subsequence T of $\overline{S}_n$, by

Theorem 2.4(*ii*), such that $c(S) = X$. In the second case, there exists a subsequence of S_{n-2} of the form

$$S = 1, 2, 1, \ldots, l, 1,$$

for some $l > 2$, starting from the first integer 1 of S_{n-2}, because of a similar reason as in Case 3, with the property that $c(S) = X$. We write $S_{n-2} = S, S''$ and we define a sequence

$$T = S'', n-1, S, S'', n$$

which is a subsequence of $\overline{S}_n$, according to Theorem 2.3(*ii*). It is obvious that

$$c(T) = c(S) \cup \{n-1, n\} = Y \cup \{n-1, n\} = X.$$

Since the first integer of S'' is 2, the sequence T satisfies all requirements. Since statement (*i*) of the Theorem, holds for $n = 3$, as we already remarked, it holds for all cases $n \geq 3$, according to the principle of mathematical induction.

(ii) This statement follows immediately from (*i*) since $S_n^R = S_n$, which together with eq. (2.13) implies that $\overline{S}_n$, considered as a cyclic sequence, is invariant for the direction of traversing (cf. Th. 2.4(*iii*)).

(iii) For $i < n$, we can take some $k > i$. Now part (*i*) of this theorem states that there is a subsequence of $\overline{S}_n$ of the form $T = i,T', k$ such that $c(T) = X$. From the definition of the transition sequence $\overline{S}_n$, we can immediately conclude that T is part of a longer subsequence $S_{i-1}, i,T', k, S_{i-1}$, and hence also of a sub- sequence S_j, i, T', k, S_j, for $j \leq i-1$. This subsequence is shorter for $j < i - 1$. The statement now follows from the relation $S_j^R = S_j$. For $i = n$, we have $X = \{n - 1\}$, and the statement follows from $S_{n-1} = S_{n-1}^R$. □

Theorem 2.8

Let $\overline{S}_n$ be the transition sequence of the standard Gray code $G(n)$. Let X be some non-empty subset of $[1, n]$, $n \geq 3$, with $|X|$ even and $i_0 := \min X$. If $i := i_0 + 1$, then

(i) there exists a subsequence $T = i, T', 1$ of $\overline{S}_n$ such that $c(T) = X$;

(ii) there exists a subsequence $T = 1, T', i$ of $\overline{S}_n$ such that $c(T) = X$.

Proof.

(*i*) We distinguish between the cases $1 \notin X$ and $1 \in X$. If $1 \notin X$, it follows that $i_0 > 1$, and hence $2 < i \leq n$. We define $Y := X \cup \{1\}$. So, $|Y|$ is odd and $\min Y = 1$. From Theorem 2.7(*ii*), it follows that there exists a subsequence $S = i, T', 2$ such that $c(S) = Y$. Hence, the subsequence $T = S, 1 = i, T', 2, 1$ satisfies the requirement.

If $1 \in X$, it follows that $i_0 = 1$, $i = 2$. We define $Y := X \setminus \{1\}$. So $|Y|$ is odd. If $2 \notin Y$, then $\min Y \geq 3$, and therefore there exists a subsequence $S = 2, S', 2$ with $c(S) = Y$, according

to Theorem 2.7(*iii*). Hence, $T := 2, S', 2, 1$ satisfies the requirements. If $2 \in X$, there exists a subsequence $S = 1, S', 1$ with $c(S) = Y$. If the first integer of S' is 2, we omit the first 1 of S giving a subsequence $T = 2, T', 1$ which satisfies all the requirements. If the first integer of S' is unequal to 2, then its last integer must be equal to 2, by Theorem 2.7(*iii*). We now omit the last 1 of T'. By applying $S_n^R = S_n$, we obtain a subsequence which satisfies (*i*) in this case.

(*ii*) This statement follows again by applying $S_n^R = S_n$. □

Theorem 2.9

Let $\overline{S}_n$ *be the transition sequence of the standard Gray code* $G(n)$ *and let* X *be some non-empty subset of* $[1, n]$, $n \geq 3$. *Let furthermore* $i_0 := \min X$ *and* $i := i_0 + 1$.

(*i*) *if* $|X|$ *is even, the number of subsequences* $T = 1, T', j$ *in* $\overline{S}_n$ *with* $c(T) = X$ *is equal to* 2^{n-j} *for* $1 < j < n$, *and equal to* 2 *for* $j = n$.

(*ii*) *if* $|X|$ *is odd, the number of subsequences* $T = i, T', j$ *in* $\overline{S}_n$ *with* $c(T) = X$ *is equal to* 2^{n-j} *for* $1 < j < n$, *and equal to* 2 *for* $j = n$.

Proof. First we remark that 2^{n-j} is equal to the number of occurrences of the integer j in $\overline{S}_n$ for $1 \leq j < n$ (*transition count* of j), and that similarly the transition count of n is equal to 2 in $\overline{S}_n$. Furthermore, it is evident that there are no two different subsequences $T^{(1)} = i, T^{(1)\prime}, j$ and $T^{(2)} = i, T^{(2)\prime}, j$ with j on the same position in $\overline{S}_n$ and the two integers i on different positions, with $c(T^{(1)}) = c(T^{(2)}) = X$, for some fixed X. Since this would imply $c(T^{(1)} \setminus T^{(2)})$ (or $c(T^{(2)} \setminus T^{(1)})$) = $\varnothing$, which violates the fact that $\overline{S}_n$ generates 2^n different codewords (cf. [33 - 36]). So, the proof comes down to showing that in either case, for each $j > i$, there corresponds a sequence $1, T', j$ (case (*i*)) or i, T', j (case(*ii*)), respectively, the contents (cf. Definition 2.1) of which is equal to X.

(*i*) Choose some integer j (> 1) in $\overline{S}_n$. Consider the set of subsequences $k, \ldots, j$ (with that particular integer j at the end) where k can be any integer to the left of j in cyclic sense. Starting from the zero codeword, any of these sequences defines a binary codeword of length n. Since $\overline{S}_n$ is the transition sequence of a complete Gray code, all these words are different. Hence, there is a k such that $c(k, \ldots, j) = X$. Since $|X|$ is even, $k = 1$ according to Theorem 2.5(*iv*).

(*ii*) Let $X = \{i_0, i_1, \ldots, i_{l-1}\}$, l odd. W.l.o.g, we may assume $i_0 < i_1 < \ldots < i_{l-1}$. First we suppose $i_0 > 1$, and hence $i > 2$ and we define $Y = \{i_0 - 1, i_1 - 1, \ldots, i_{l-1} - 1\}$. From Theorem 2.3(*ii*) it follows that there is a one-one correspondence between subsequences $T = i, T', j$, in $\overline{S}_n$ with

$c(T) = X$ and subsequences $S = i - 1, S', j - 1$ in $\overline{S}_{n-1}$ with $c(S) = Y$. Proceeding in this way, we conclude that there is one-one correspondence between subsequences $T = i, T', j$ in $\overline{S}_n$ and subsequences $S'' = i - i_0, S''', j - i_0 = 1, S''', j - i_0$ in $\overline{S}_{n-i_0}$ with $c(S'') = \{i_1 - i_0, \ldots, i_{l-1} - i_0\}$. From part (*i*) of this proof it follows that the total number of such sequences is equal to

$$2^{(n-i_0)-(j-i_0)} = 2^{n-j}$$

if $j < n$, and equal to 2 if $j = n$. □

Example 2.2

The transition sequence of $\overline{S}_5$ is equal to

1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5.

We define $X = \{1, 2, 5\}$ and so $i_0 = 1$ and $i = 2$. There are two subsequences $T = 2, T', 5$ with $c(T) = X$, i.e the two sequences 2, 1, 5; there are also two sequences $T = 2, T', 4$ with $c(T) = X$, i.e. the sequences

2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4 (twice).

Next, we consider $X = \{1\}$, hence $i_0 = 1$, $i = 2$, and we take $j = 3$. The following subsequences $T = 2, T', 3$ have contents $c(T) = \{1\}$:

2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, (twice)

2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, (twice)

Finally, we consider $X = \{2, 3\}$, so $i_0 = 2$ and $i = 3$, and we take $j = 3$. The following subsequences $T = 1, T', 3$ have contents $c(T) = \{2, 3\}$:

1, 2, 1, 3 (four times).

The previous example might suggest that all subsequences $T = i, T', j$ for fixed values of i and j which satisfy $c(T) = X$ for some fixed set X, have the same length. This is not true in general. The next example will illustrate this.

Example 2.3

According to Theorem 2.9, there are $2^{5-3} = 4$ subsequences $T = 2, T', 3$ of $\overline{S}_5$ which satisfy $c(T) = \{1, 3, 5\}$. These subsequences are

2, 1, 5, 1, 2, 1, 3, (twice)

of length 7, and

2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, (twice)

of length 23.

2.4 *DP*-Codes, <*m*, *n*>- Codes and Snakes

In [25], Preparata and Nievergelt discussed the notion of *difference-preserving code* (shortly, *DP*-code), being an ordered list of integers, coded as binary words of length n, satisfying the following two properties:

(*i*) (*nearness condition*) the list distance between two words is equal to their Hamming distance, as long as the list distance does not exceed a certain *threshold* or *spread* m;

(*ii*) (*separability condition*) if the list distance exceeds m, so does the Hamming distance.

The *range* L of the code is the number of words in the list. If one considers the list as a cyclic list and if the list distance between two words is taken cyclicly, i.e, identifying the first word and the L-th word (cf. (2.11)), then one can speak of *cyclic* DP-codes.

Some authors (cf. [10]) slightly modify the condition (*ii*) by

(*ii'*) if the list distance is at least m, so is the Hamming distance.

A more general type of ordered code was introduced by Evdokimov. In [10], he studied codes satisfying condition (*i*), not bothering about condition (*ii*), We shall call such a code a *distance-preserving* <*m*, *n*>-code, or simply an <*m*, *n*>-code. This class includes the well-known *snake-in-the-box* codes which will be discussed in the next section. It will be clear that $1 \le m \le n$. It will also be clear that a cyclic Gray code $G(n)$ is by definition a cyclic <1, *n*>-code. Since the same bit will not be changed immediately again, it is even a cyclic <2, *n*>-code.

More precisely, an <*m*, *n*>-code is an ordered code

$$F(n) = \mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{L-1}.$$

of words in Q_n that satisfies the condition

$$|i - j| \le m \Rightarrow d(\mathbf{w}_i, \mathbf{w}_j) = |i - j|, \qquad 0 \le i, j \le L - 1.$$

This code is called *complete* if $L = 2^n$ and it is *cyclic* if

$$l(\mathbf{w}_i, \mathbf{w}_j) \le m \Rightarrow d(\mathbf{w}_i, \mathbf{w}_j) = l(\mathbf{w}_i, \mathbf{w}_j), \qquad 0 \le i, j \le L - 1,$$

where the cyclic list distance $l(\mathbf{w}_i, \mathbf{w}_j)$ is defined in (2.12). One could say that $F(n)$ preserves the distance between pairs of integers from $[0, L - 1]$, for distances up to m. Moreover, $F(n)$ is called *symmetric* if it has a transition sequence of the form

$$t_1, t_2, \ldots, t_K, t_1, t_2, \ldots, t_K. \tag{2.17}$$

where $L = 2K$.

Example 2.4

In [39], one can find an example of a complete cyclic <4, 5>-code.

0. 00000	8. 00011	16. 00110	24. 00101
1. 00100	9. 00001	17. 00010	25. 00111
2. 10100	10. 10001	18. 10010	26. 10111
3. 10110	11. 10101	19. 10000	27. 10011
4. 11110	12. 11101	20. 11000	28. 11011
5. 11111	13. 11100	21. 11001	29. 11010
6. 01111	14. 01100	22. 01001	30. 01010
7. 01011	15. 01110	23. 01101	31. 01000.

In this thesis, the notation <m, n>-code will always stand for a *complete cyclic* <m, n>-code.

Next we define the related notion of a *spread-δ circuit* (or shortly, circuit).

Definition 2.2

A list of binary words of length n is called a spread-δ circuit if and only if the following two conditions are satisfied for all words x and y of this list

(a) $l(\mathbf{x},\mathbf{y}) = 1$ *implies* $d(\mathbf{x},\mathbf{y}) = 1$;

(b) $d(\mathbf{x},\mathbf{y}) < \delta$ *implies* $d(\mathbf{x},\mathbf{y}) = l(\mathbf{x},\mathbf{y})$.

This definition was given by Paterson and Tuliani (cf. [24]). Preparata and Nievergelt in [25] give the following definition of spread-δ circuits that at first glance slightly differs from the definition above. We shall prove that both definitions are equivalent.

Definition 2.3

A list of binary words of length n is called a spread-δ circuit, if only if the following two conditions are satisfied for all words **x** *and* **y** *of this list:*

(a') $l(\mathbf{x},\mathbf{y}) \le \delta$ *implies* $d(\mathbf{x},\mathbf{y}) = l(\mathbf{x},\mathbf{y})$;

(b') $l(\mathbf{x},\mathbf{y}) > \delta$ *implies* $d(\mathbf{x},\mathbf{y}) \ge \delta$.

Theorem 2.10

The two Definitions 2.2 *and* 2.3 *for spread-δ circuits are equivalent.*

Proof. Assume that the conditions of Definition 2.2 hold for some list C.

(a') Let $l(\mathbf{x},\mathbf{y}) < \delta$. By applying ($a$) $l(\mathbf{x}, \mathbf{y})$ times, it follows that $d(\mathbf{x}, \mathbf{y}) \le l(\mathbf{x}, \mathbf{y}) < \delta$. But then, because of ($b$), we have $d(\mathbf{x},\mathbf{y}) = l(\mathbf{x},\mathbf{y})$. Let $l(\mathbf{x},\mathbf{y}) = \delta$. Then $d(\mathbf{x},\mathbf{y}) \le \delta$, again by ($a$). If $d(\mathbf{x},\mathbf{y}) < \delta$, we would have $\delta > d(\mathbf{x},\mathbf{y}) = l(\mathbf{x},\mathbf{y}) = \delta$ from (b), which is false. So, $d(\mathbf{x},\mathbf{y}) = \delta = l(\mathbf{x},\mathbf{y})$. We conclude that $l(\mathbf{x},\mathbf{y}) \le \delta$ implies $d(\mathbf{x},\mathbf{y}) = l(\mathbf{x},\mathbf{y})$.

(b') Let $l(\mathbf{x},\mathbf{y}) > \delta$. If $d(\mathbf{x},\mathbf{y}) < \delta$, then because of ($b$), $\delta < l(\mathbf{x},\mathbf{y}) = d(\mathbf{x},\mathbf{y}) < \delta$, which is false.

So, $d(\mathbf{x},\mathbf{y}) \geq \delta$.

We conclude that the conditions of Definition 2.3 also hold for C.

Assume next that the conditions of Definition 2.3 hold for some list C.

(a) It follows immediately that $l(\mathbf{x},\mathbf{y}) = 1$ implies $d(\mathbf{x},\mathbf{y}) = 1$, i.e. condition ($a$) of Definition 2.2 is satisfied.

(b) Now let $d(\mathbf{x},\mathbf{y}) < \delta$. The inequality $l(\mathbf{x},\mathbf{y}) > \delta$ implies, by (b'), that $d(\mathbf{x},\mathbf{y}) \geq \delta$, which is false. So $l(\mathbf{x},\mathbf{y}) \leq \delta$ and it follows from condition (b') that $l(\mathbf{x},\mathbf{y}) = d(\mathbf{x},\mathbf{y})$.

Therefore, the conditions of the Definition 2.2 are also satisfied. □

A spread-2 circuit is called *a snake-in-the-box code*, or simply a *snake*. *DP*-codes, <m, n>-codes and snakes are all special Gray codes (cf. e.g. [39]). The <4, 5>-code in Example 2.4 is clearly not a snake, nor a *DP*-code because many pairs of non-consecutive words in the code, e.g. the 6-th and the 15-th words, are at distance $1 < 2$.

More generally, a snake in a graph is an *induced* cycle in that graph, that is a simple cycle with no chords. A chord of a cycle $\mathcal{S}$ is an edge which connects two non-consecutive vertices of $\mathcal{S}$. Only if the graph is the hypercube graph Q_n, one speaks of a snake-in-the-box code. More precisely, suppose

$$\mathcal{S} = \mathbf{w}_0, \mathbf{w}_1, ..., \mathbf{w}_{L-1} \tag{2.18}$$

is a list of L words in Q_n. Then $\mathcal{S}$ is a snake-in-the-box code, or simply a *snake*, if for every integer $i, j \in [0, L-1]$, we have

$$d[\mathbf{w}_i, \mathbf{w}_{i+1}]=1 \tag{2.19}$$

and

$$d[\mathbf{w}_i, \mathbf{w}_j]=1 \Rightarrow l[\mathbf{w}_i, \mathbf{w}_j]=1, \tag{2.20}$$

where $\mathbf{w}_L$ is identified with $\mathbf{w}_0$.

We call a snake in Q_n *symmetric* if it has a transition sequence of the form (2.17). Kautz [15] calls such a snake a *natural code*. The first half of the snake can be derived from the second half by executing the same coordinate changes.

3. A Construction of Snake-in-the-box Codes

In this chapter, we start by discussing a method for the construction of *snake-in-the-box codes* - briefly called *snakes* - due to Paterson and Tuliani. Their method makes use of so-called *necklaces* which are a certain kind of cyclic constant-weight codes. Actually, Paterson and Tuliani present in [24] two different constructions for snakes based on such necklaces, the second one of which is not proved to be correct. In Section 3.1, we discuss the two constructions of snakes by Paterson and Tuliani.

In Section 3.2, we generalize the notion of necklaces and in Section 3.3, we introduce a third construction which generalizes the two constructions in [24], and we prove its correctness. In particular, we construct the currently known longest snake for $n = 8$ that was also provided by Paterson and Tuliani in [24].

3.1. Snakes Based on Necklaces

A *necklace* is defined by Patterson and Tuliani in [24] as a class of a special equivalence relation in Q_n. The equivalence relation is defined in the following way. Two vectors **v** and **w** of Q_n are in the same necklace if and only if **w** can be obtained from **v** by shifting the components of **v** over one or more positions in cyclic sense.

We can represent a necklace by a matrix the rows of which are all vectors in the equivalence class it stands for. If a vector coincides with itself when shifted (cyclically) over k (> 1) positions, we say that the corresponding necklace is of *period* k, if k is minimal with respect to this property. If the period of a necklace, is equal to n, we shall speak of a *full-period* necklace . In the sequel, we always mean a full-period necklace when we write 'necklace', unless stated otherwise.

The 6×6 matrix in Figure 1 represents a necklace generated by its first row vector (1,1,1,0,1,0). We can also represent the necklace by merely this row: [111010]. We shall call such a matrix a *necklace matrix*. Since all rows of a necklace have the same weight, we define the *weight of a necklace* N, wt(N), as the weight of its rows.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Fig. 3.1 A Necklace Matrix

Since each row of the matrix in our example has four 1's, we can assign a block (r, s, t, u) to each of them such that r, s, t and u indicate the positions of these 1's. Thus, to the first row we assign the block $B_1 = (1, 2, 3, 5)$, the second row corresponds to the block $B_2 = (2, 3, 4, 6)$, etc.

Furthermore, we introduce a *distance distribution matrix* of a necklace, the i,j-entry of which is the Hamming distance between row i and row j. As one can easily verify, the matrix in Fig.3.2 is the distance distribution matrix of the necklace represented by the matrix of Fig. 3.1.

Since the whole necklace matrix is determined by its first row, the overall distance distribution is already determined by the first row of the distance distribution matrix. Therefore, we may represent the distribution matrix of Figure 3.2 by its first row [0 4 2 4 2 4].

$$\begin{pmatrix} 0 & 4 & 2 & 4 & 2 & 4 \\ 4 & 0 & 4 & 2 & 4 & 2 \\ 2 & 4 & 0 & 4 & 2 & 4 \\ 4 & 2 & 4 & 0 & 4 & 2 \\ 2 & 4 & 2 & 4 & 0 & 4 \\ 4 & 2 & 4 & 2 & 4 & 0 \end{pmatrix}$$

Fig. 3.2 A Distance Distribution Matrix of a Necklace

Let $\mathbf{w} = (w_0, w_1, \ldots, w_{n-1})$ be a binary word of length n. We introduce the notation σ for the operator which accomplishes a cyclic shift over one position to the left. Hence,

$$\sigma\mathbf{w} = (w_1, w_2, \ldots, w_{n-1}, w_0)$$

and more generally,

$$\sigma^l\,\mathbf{w} = (w_l, w_{l+1}, \ldots, w_{n-1}, w_0, \ldots, w_{l-1})$$

for any nonnegative integer l.

Now, let N_1 and N_2 be two necklaces of length n and let $\mathbf{w}_1$ and $\mathbf{w}_2$ be fixed vectors of N_1 and N_2, respectively. The distance between N_1 and N_2 is defined as

$$d(N_1, N_2) = \min_{0 \le l < n} d(\mathbf{w}_1, \sigma^l\,\mathbf{w}_2).$$

It will be obvious that the distance between N_1 and N_2 is independent of the choice of the vectors $\mathbf{w}_1$ and $\mathbf{w}_2$.

We remark that if the distance between N_1 and N_2 is equal to d, there might be more than one value l such that

$$d(\mathbf{w}_1, \sigma^l\mathbf{w}_2) = d. \tag{3.1}$$

In case that there is a unique value l defining the distance between two necklaces, one can specify the notion of distance as follows. The necklaces N_1 and N_2 are *uniquely at distance d*

with respect to k, if l in (3.1) is uniquely determined and if for every $l' \neq l$, $0 \leq l' < n$, one has

$$d(\mathbf{w}_1, \sigma^{l'}\mathbf{w}_2) \geq k.$$

From the uniqueness of l, it follows immediately that $k > d$.

Example 3.1.

For $n = 7$, we define the following necklaces

$$N_0 = [1100000], N_1 = [1110000], N_2 = [1111000], N_3 = [1111100].$$

These necklaces are all full-period of length 7. Written completely as 7×7 matrices, they have the form

$$N_0 = \begin{pmatrix} 1&1&0&0&0&0&0 \\ 1&0&0&0&0&0&1 \\ 0&0&0&0&0&1&1 \\ 0&0&0&0&1&1&0 \\ 0&0&0&1&1&0&0 \\ 0&0&1&1&0&0&0 \\ 0&1&1&0&0&0&0 \end{pmatrix}, \quad N_1 = \begin{pmatrix} 1&1&1&0&0&0&0 \\ 1&1&0&0&0&0&1 \\ 1&0&0&0&0&1&1 \\ 0&0&0&0&1&1&1 \\ 0&0&0&1&1&1&0 \\ 0&0&1&1&1&0&0 \\ 0&1&1&1&0&0&0 \end{pmatrix},$$

$$N_2 = \begin{pmatrix} 1&1&1&1&0&0&0 \\ 1&1&1&0&0&0&1 \\ 1&1&0&0&0&1&1 \\ 1&0&0&0&1&1&1 \\ 0&0&0&1&1&1&1 \\ 0&0&1&1&1&1&0 \\ 0&1&1&1&1&0&0 \end{pmatrix}, \quad N_3 = \begin{pmatrix} 1&1&1&1&1&0&0 \\ 1&1&1&1&0&0&1 \\ 1&1&1&0&0&1&1 \\ 1&1&0&0&1&1&1 \\ 1&0&0&1&1&1&1 \\ 0&0&1&1&1&1&0 \\ 0&1&1&1&1&0&0 \end{pmatrix}.$$

If we define $\mathbf{w}_0 = (1,1,0,0,0,0,0) \in N_0$, $\mathbf{w}_1 = (1,1,1,0,0,0,0) \in N_1$, $\mathbf{w}_2 = (1,1,1,1,0,0,0) \in N_2$ and $\mathbf{w}_3 = (1,1,1,1,1,0,0) \in N_3$, we can easily verify that

$$d(\mathbf{w}_i, \sigma\mathbf{w}_{i+r}) \geq r$$

for $0 \leq i < i + r \leq 4$. It can also be verified easily that

$$d(N_i, N_{i+r}) = r$$

for the same values of i and r. However, N_i and N_{i+r} are not uniquely at distant r from each other. E.g. $d(\mathbf{w}_0, \sigma^1\mathbf{w}_1) = d(\mathbf{w}_0, \sigma\mathbf{w}_1) = 1$ and $d(\mathbf{w}_0, \sigma^0\mathbf{w}_1) = d(\mathbf{w}_0, \mathbf{w}_1) = 1$, so the value of l in (3.1) is not unique.

Example 3.2.

Consider the necklaces

$$N_0 = [1100000], N_1 = [1101000], N_2 = [1101100], N_3 = [1111100].$$

or written as complete matrices

$$N_0 = \begin{pmatrix} 1&1&0&0&0&0&0\\ 1&0&0&0&0&0&1\\ 0&0&0&0&0&1&1\\ 0&0&0&0&1&1&0\\ 0&0&0&1&1&0&0\\ 0&0&1&1&0&0&0\\ 0&1&1&0&0&0&0 \end{pmatrix}, \quad N_1 = \begin{pmatrix} 1&1&0&1&0&0&0\\ 1&0&1&0&0&0&1\\ 0&1&0&0&0&1&1\\ 1&0&0&0&1&1&0\\ 0&0&0&1&1&0&1\\ 0&0&1&1&0&1&0\\ 0&1&1&0&1&0&0 \end{pmatrix},$$

$$N_2 = \begin{pmatrix} 1&1&0&1&1&0&0\\ 1&0&1&1&0&0&1\\ 0&1&1&0&0&1&1\\ 1&1&0&0&1&1&0\\ 1&0&0&1&1&0&1\\ 0&0&1&1&0&1&1\\ 0&1&1&0&1&1&0 \end{pmatrix}, \quad N_3 = \begin{pmatrix} 1&1&1&1&1&0&0\\ 1&1&1&1&0&0&1\\ 1&1&1&0&0&1&1\\ 1&1&0&0&1&1&1\\ 1&0&0&1&1&1&1\\ 0&0&1&1&1&1&1\\ 0&1&1&1&1&1&0 \end{pmatrix}.$$

Here, all pairs N_i and N_{i+1} are uniquely at distance 1 with respect to 2. Moreover, $d(N_i, N_{i+r}) = r$ for $0 \le i < i + r \le 4$. But not all these distances are uniquely determined. E.g. $d(\mathbf{w}_0, \sigma^0\mathbf{w}_2) = d(\mathbf{w}_0, \sigma^3\mathbf{w}_2) = 2$.

In [24], Paterson and Tuliani present the following construction, (called Construction 1) for spread-δ circuits based on the existence of a set of necklaces which satisfy certain conditions.

Theorem 3.1 (Paterson and Tuliani; *Construction* 1)

Let $t \ge \delta$ and suppose that $N_0, N_1, \ldots, N_{t-1}$ are distinct, full-period necklaces of length n, such that for every i with $0 \le i < t$, and all indices taken modulo t, the following conditions hold:

(i) *N_i and N_{i+r} are uniquely at distance r with respect to δ, for $0 \le r < \delta$;*

(ii) *N_i and N_{i+r} are at distance $\ge \delta$ for $\delta \le r \le t - \delta$.*

Then there exist words $\mathbf{w}_i \in N_i$ with $d(\mathbf{w}_i, \mathbf{w}_{i+1}) = 1$, $0 \le i < t$, and an integer l, $0 \le l < n$, with $d(\mathbf{w}_{t-1}, \sigma^l\mathbf{w}_0) = 1$. Furthermore, there is an integer m such that the list C obtained by concatenation of the rows

$$\begin{array}{cccc} \mathbf{w}_0, & \mathbf{w}_1, & \mathbf{w}_2, & \ldots, \ \mathbf{w}_{t-1} \\ \sigma^l\mathbf{w}_0, & \sigma^l\mathbf{w}_1, & \sigma^l\mathbf{w}_2, & \ldots, \ \sigma^l\mathbf{w}_{t-1} \\ \ldots, & \ldots, & \ldots, & \ldots, \\ \sigma^{(m-1)l}\mathbf{w}_0, & \sigma^{(m-1)l}\mathbf{w}_1, & \sigma^{(m-1)l}\mathbf{w}_2, & \ldots, \ \sigma^{(m-1)l}\mathbf{w}_{t-1} \end{array}$$

is a spread-δ circuit with mt words of length n, where m is defined as

$$m := \frac{n}{\gcd(l, n)}.$$

Example 3.3.

We take again the first three necklaces N_0 = [1100000], N_1 = [1101000], N_2 = [1101100] from the previous example, but we choose N_3 = [1011000] that gives rise to the following matrix:

$$N_3 = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Now, we have that N_i and N_{i+1} are uniquely at distance 1 with respect to 2, for all i mod 4. Furthermore, N_i and N_{i+2} are at distance 2 for all i mod 4, although not uniquely.

According to the algorithm of Paterson and Tuliani, we are able now to construct a spread-2 circuit (snake-in-the-box) of length 28. Since N_i and N_{i+1} are uniquely at distance 1 for all i mod 4, this snake is uniquely determined. Written as a concatenation of rows, it is given by

1100000	1101000	1101100	0101100
0001100	0001101	1001101	1000101
1000001	1010001	1011001	1011000
0011000	0011010	0011011	0001011
0000011	0100011	0110011	0110001
0110000	0110100	0110110	0010110
0000110	1000110	1100110	1100010

The transition sequence of this snake is obtained by concatenation of the rows

4	3	7	6
1	7	4	3
5	4	1	7
2	1	5	4
6	5	2	1
3	2	6	5
7	6	3	2

We remark that the above transition sequence is determined already by the first row. Every time a next row can be obtained by adding 4 mod 7 to each entry of the previous row. The entries of the first row are determined by the words $\mathbf{w}_0$, $\mathbf{w}_1$, $\mathbf{w}_2$ and $\mathbf{w}_3$.

Corollary 3.2

Let $\mathcal{S}$ *be the snake with word length n obtained by Construction* 1 *that uses t necklaces, and let* T_{mt} *be its transition sequence with elements* $e_1, e_2, \ldots, e_t, e_{t+1}, \ldots, e_{2t}, e_{2t+1}, \ldots, e_{mt}$. *If l is the only nonzero integer satisfying* $d(\mathbf{w}_{t-1}, \sigma^l \mathbf{w}_0) = 1$, *then the following relation holds*

$$e_{rt+s} = e_s + rl \bmod n$$

for $r = 0, 1, \ldots, m-1$ *and* $s = 1, 2, \ldots, t$.

Proof. Any pair of words $(\mathbf{w}_{(r+1)t+s}, \mathbf{w}_{(r+1)t+s+1})$ in the r–th row originates from the pair $(\mathbf{w}_{rt+s}, \mathbf{w}_{rt+s+1})$ in the $(r-1)$-th row by a left shift over l positions. Therefore, the equality holds for all $s < t$. For $s = t$, the relation is true for a similar reason. □

Example 3.4

The following six necklaces with word length $n = 6$

[110001], [110101], [110111], [100111], [100110], [000110]

do not satisfy the conditions in Theorem 3.1. In fact, there is a pair of necklaces, the first and the last one, that are not uniquely at distance 1. Moreover, the first and the fourth necklace are uniquely at distance 1, although their list distance is 3. Therefore, the scheme outlined in Theorem 3.1 (with two possible values $l = 2$ or $l = 3$) produces a Gray code, that is not a snake, of range

$$N = mt = \frac{6}{\gcd(l,6)} 6 = 36/l.$$

By putting a bit 0 to the right of the last position of every word representing the necklaces, we get the following six necklaces with word length 7 that satisfy all conditions of Theorem 3.1.

[1100010], [1101010], [1101110], [1001110], [1001100], [0001100].

Applying Construction 1 yields the following snake of length 42.

1100010, 1101010, 1101110, 1001110, 1001100, 0001100,
0101100, 0101101, 1101101, 1101001, 1001001, 1000001,
1000101, 1010101, 1011101, 0011101, 0011001, 0011000,
1011000, 1011010, 1011011, 1010011, 0010011, 0000011,
0001011, 0101011, 0111011, 0111010, 0110010, 0110000,
0110001, 0110101, 0110111, 0100111, 0100110, 0000110,
0010110, 1010110, 1110110, 1110100, 1100100, 1100000.

Among all snakes with word length 7 that can be obtained by Construction 1, this snake is maximal. Any longer snake obtained by Construction 1 should be generated by at least seven necklaces. This implies that the length of the snake would be at least 49, exceeding the maximal length of a snake with word length 7, which is known to be 48 (see [17]).

Clearly, every vector in a necklace N has the same weight. We denote this weight by wt(N). The next construction is also due to Patterson and Tuliani, and can be found in [24].

Theorem 3.3 (Paterson and Tuliani; *Construction* 2)

Let $t \geq 2$, *and assume that* $N_0 = [\mathbf{w}_0]$, $N_1 = [\mathbf{w}_1]$, …, $N_{t-1} = [\mathbf{w}_{t-1}]$ *are distinct, full-period* t *necklaces of length* n, *such that*

(i) $wt(N_0) = 2$ *and* $wt(N_{t-1}) = n - 2$;

(ii) $3 \leq wt(N_i) \leq n - 3$, for $1 \leq i < t - 1$;

(iii) N_i *and* N_{i+1} *are uniquely at distance 1 with respect to* 2, *for* $0 \leq i < t - 1$;

(iv) $d(N_i, N_{i+r}) \geq 2$, *for* $0 \leq i < t$ *and* $2 \leq r \leq t - 2$, *where indices are taken mod* t.

(v) each word $\mathbf{w}_{i+1}$, , $0 \leq i < t - 1$, *is at distance* 1 *from its predecessor,* $\mathbf{w}_i$, *and moreover the two* 1*-bits of* $\mathbf{w}_0 \in N_0$ *are at two positions labeled with different parities, so are the two* 0*-bits of* $\mathbf{w}_{t-1}. \in N_{t-1}$.

Then there exist two odd integers $l, l' \in [1, n/2 - 1]$ *satisfying* $d(\mathbf{w}_{t-1}, \sigma^l \mathbf{w}_{t-1}) = 2$ *and* $d(\sigma^l \mathbf{w}_0, \sigma^{l+l'} \mathbf{w}_0) = 2$, *two words* $\mathbf{v}_0$ *and* $\mathbf{v}_1$ *satisfying* $wt(\mathbf{v}_0) = n - 1$, $d(\sigma^l \mathbf{w}_{t-1}, \mathbf{v}_0) = d(\mathbf{w}_{t-1}, \mathbf{v}_0) = 1$, $wt(\mathbf{v}_1) = 1$, $d(\sigma^l \mathbf{w}_0, \mathbf{v}_1) = d(\sigma^{l+l'} \mathbf{w}_0, \mathbf{v}_1) = 1$. *From these words the following scheme (cf.* [24]*) is produced*

$$
\begin{array}{ccccc}
\mathbf{w}_0, & \mathbf{w}_1, & \mathbf{w}_2, & \ldots, & \mathbf{w}_{t-1} \\
 & & & & \mathbf{v}_0, \\
\sigma^l \mathbf{w}_0, & \sigma^l \mathbf{w}_1, & \sigma^l \mathbf{w}_2, & \ldots, & \sigma^l \mathbf{w}_{t-1}, \\
\mathbf{v}_1, & & & & \\
\sigma^{l+l'} \mathbf{w}_0, & \sigma^{l+l'} \mathbf{w}_1, & \sigma^{l+l'} \mathbf{w}_2, & \ldots, & \sigma^{l+l'} \mathbf{w}_{t-1}, \\
 & & & & \mathbf{v}_2, \\
\sigma^{2l+l'} \mathbf{w}_0, & \sigma^{2l+l'} \mathbf{w}_1, & \sigma^{2l+l'} \mathbf{w}_2, & \ldots, & \sigma^{2l+l'} \mathbf{w}_{t-1}, \\
\mathbf{v}_3 & & & & \\
\sigma^{2l+2l'} \mathbf{w}_0, & \sigma^{2l+2l'} \mathbf{w}_1, & \sigma^{2l+2l'} \mathbf{w}_2, & \ldots, & \sigma^{2l+2l'} \mathbf{w}_{t-1}, \\
 & & & & \mathbf{v}_4, \\
\ldots & \ldots & \ldots & \ldots, & \ldots \\
\sigma^{ml+(m-1)l'} \mathbf{w}_0, & \sigma^{ml+(m-1)l'} \mathbf{w}_1, & \sigma^{ml+(m-1)l'} \mathbf{w}_2, & \ldots, & \sigma^{ml+(m-1)l'} \mathbf{w}_{t-1}, \\
\mathbf{v}_{2t+1}, & & & &
\end{array}
$$

In the above theorem $\mathbf{v}_2 = \sigma^{l+l'} \mathbf{v}_0$ is a word with distance 1 w.r.t. $\sigma^l \mathbf{w}_0$ and $\sigma^{l+l'} \mathbf{w}_0$, $\mathbf{v}_3 = \sigma^{l+l'} \mathbf{v}_1$ is a word with distance 1 w.r.t. $\sigma^{l+l'} \mathbf{w}_{t-1}$ and $\sigma^{2l+l'} \mathbf{w}_{t-1}$.

In the original Construction 2, Paterson and Tuliani in [24] did not state about the odd parity of the integers l and l', neither did they present sufficient or necessary conditions for the correctness of the procedure. Neither the existence of two words $\mathbf{v}_0$ and $\mathbf{v}_1$ nor the conditions imposed on the two 1-bits and the two 0-bits of $\mathbf{w}_0$ and of $\mathbf{w}_{t-1}$, respectively, was explicitly stated by them. They even did not prove that the above procedure always produces a snake,

In Section 3.3, we shall show that the above Construction 2 is a special case of an adapted Construction 1 (Theorem 3.1) that will be stated in Theorem 3.7. This implies that the Construction 2 of Theorem 3.3 is valid, and does produce a snake once one can show that the Construction 3 is indeed valid and therefore, as one of its special cases, the scheme given in Theorem 3.3 (Construction 2) does produce a snake.

Anyway, it will be clear that a necessary condition for the production of a spread-2 circuit, is that n is even. In the two examples presented in [24], the values of n are 8 and 10, respectively. Our next example shows that for odd values of n, the resulting sequence is a spread-2 circuit, sometimes called *open snake*, but not a spread-2 circuit.

Example 3.5

We consider the set of necklaces of Example 3.2. These necklaces satisfy the conditions *(i) – (iv)*. Applying the algorithm yields the following sequence of words, which clearly is an open snake.

1100000	1101000	1101100	1111100
			1111101
1000001	1010001	1011001	1111001
0000001			
0000011	0100011	0110011	1110011
			1110111
0000110	1000110	1100110	1100111
0000100			
0001100	0001101	1001101	1001111
			1011111
0011000	0011010	0011011	0011111
0010000			
0110000	0110100	0110110	0111110

Corollary 3.4

Let $\mathcal{S}$ *be the snake with word length n obtained by Construction 2 that uses t necklaces, and let T be its transition sequence with elements* $e_1, e_2, \ldots, e_t, e_{t+1}, \ldots, e_{2t}, e_{2t+1}, \ldots e_{mt}$. *Suppose k and l are two nonzero integers satisfying*

$$d(\mathbf{w}_0, \sigma^l \mathbf{w}_0) = 2 \qquad \textit{and} \qquad d(\mathbf{w}_{t-1}, \sigma^k \mathbf{w}_{t-1}) = 2,$$

which exist because of the conditions of Construction 2. If n is the word length, then the following two relations hold (mod n)

(i) $e_{t+1+s} = e_{t-s} + k$, *for* $s = 1, 2, \ldots, t-1$;

(ii) $e_{(r+2)(t+1)+s} = e_{r(t+1)+s} + k + l$, *for* $r = 0, 1, 2, \ldots$, *and* $s = 1, 2, 3, \ldots, t+1$.

Proof. We arrange the elements of T in groups of t+1 consecutives integers, according to the scheme below.

$$
\begin{array}{cccccccc}
 & & e_1 & e_2 & \ldots, & e_{(t+1)-2} & e_{(t+1)-1} & e_{t+1} \\
e_{2(t+1)} & e_{2(t+1)-1} & e_{2(t+1)-2} & e_{2(t+1)-3} & \ldots, & e_{(t+1)+1} & & \\
 & & e_{2(t+1)+1} & e_{2(t+1)+2} & \ldots, & e_{3(t+1)-2} & e_{3(t+1)-1} & e_{3(t+1)} \\
e_{4(t+1)} & e_{4(t+1)-1} & e_{4(t+1)-2} & e_{4(t+1)-3} & \ldots, & e_{3(t+1)+1} & & \\
 & \cdots & \cdots & \cdots & \cdots & \cdots & &
\end{array}
$$

(i) We first prove equation *(i)* that relates the first two rows. In the first and second group (i.e. the first and second row) of $t + 1$ integers above, we temporarily exclude the last two consecutive integers $e_{(t+1)-1}$, $e_{(t+1)}$ and $e_{2(t+1)-1}$, $e_{2(t+1)}$ respectively. Notice that these integers are determined by two words that are neighbors of the special words $\mathbf{w}_t$ and $\mathbf{w}_{2t+1}$, respectively. We see that both the first and the second row (excluding e_t, e_{t+1}, $e_{2(t+1)-1}$ and $e_{2(t+1)}$) play exactly the same role as the corresponding rows in Construction 1, except that the indices of the second row in this construction are in opposite order to those in Construction 1. By appropriately renaming the indices in the equation of Theorem 3.2, we get the first equation.

(ii) The second equation relates the r-th row to the (r+2)-th row in the scheme. It is sufficient to prove this equation in the case r is odd. In fact, both odd and even case consider a pair of rows that behave exactly the same, except that their indices are increasing in opposite order, while the shifts from one row to the other have different values.

When r is odd, we can use a similar argument as in part *(i)*, to derive an equivalent equation that relates the first t–1 integers in the r-th row and the corresponding $t - 1$ integers in the (r+1)-st row. Going from the (r+1)-st row to the (r+2)-nd row, we do the same thing, but now by shifting over l positions. Thus we get $e_{(r+2)(t+1)+s} = e_{r(t+1)+s} + k + l$ for $s = 1, 2, \ldots, t-1$.

The special cases $s = t$ and $s = t$+1 are treated by considering the special words $\mathbf{w}_{r(t+1)-1}$, where r is odd, which are from the same necklace by Construction 2. In this case, the pair of consecutive integers $(e_{r(t+1)-1}, e_{r(t+1)})$, which originates from the triple of words $\mathbf{w}_{r(t+1)-2}, \mathbf{w}_{r(t+1)-1}$ and $\mathbf{w}_{r(t+1)}$, corresponds to the pair of integers $(e_{(r+2)(t+1)-1}, e_{(r+2)(t+1)})$, which originates from the triple $\mathbf{w}_{(r+2)(t+1)-2}, \mathbf{w}_{(r+2)(t+1)-1}$ and $\mathbf{w}_{(r+2)(t+1)}$. This latter triple is a result of translating the former triple first over k positions followed by a translation over l positions. Thus, the pair $(e_{(r+2)(t+1)-1}, e_{(r+2)(t+1)})$ is also a translation over $k + l$ positions of the pair $(e_{r(t+1)-1}, e_{r(t+1)})$. Therefore, the second equation has been completely proved now. □

3.2. Generalized Necklaces

We can generalize the method of Paterson and Tuliani in the following way. A necklace of word length n can be seen as an orbit in $V := GF(2)^n$ of the cyclic permutation σ of the symmetric group S_n acting on the vectors of V. Instead of σ, we can choose some other permutation π of S_n. The *orbits* of π will be denoted by O_1, O_2, … . For these orbits, one can define the notions of distance and uniquely at distance δ in the same way as we did for necklaces.

Let O_1, O_2, …, O_{t-1} be a set of orbits of π, all of which have the same length. If furthermore, these orbits satisfy the conditions of Construction 1, one can construct a spread-δ circuit again.

Example 3.6

We take the words $\mathbf{w}_0$ = 110000001, $\mathbf{w}_1$ = 110100001, $\mathbf{w}_2$ = 110110001, $\mathbf{w}_3$ = 010110001. Actually, these are the words $\mathbf{w}_0$, $\mathbf{w}_1$, $\mathbf{w}_2$,$\mathbf{w}_3$ from Example 3.3, extended with 01. We now apply the permutation

$$\pi = (1,2)(3,4,5,6,7,8,9)$$

(remember that coordinates are numbered from right to left, e.g. π(110000001) = 100000110), yielding the following generalized necklaces which consist of 14 words.

01. 110000001,	110100001,	110110001,	010110001,
02. 100000110,	101000110,	101100110,	101100010,
03. 000001101,	010001101,	011001101,	011000101,
04. 000011010,	100011010,	110011010,	110001010,
05. 000110001,	000110101,	100110101,	100010101,
06. 001100010,	001101010,	001101110,	000101110,
07. 011000001,	011010001.	011011001,	001011001.
08. 110000010,	110100010,	110110010,	010110010,
09. 100000101,	101000101,	101100101,	101100001,
10. 000001110,	010001110,	011001110,	011000110,
11. 000011001,	100011001,	110011001,	110001001,
12. 000110010,	000110110,	100110110,	100010110,
13. 001100001,	001101001,	001101101,	000101101,
14. 011000010,	011010010.	011011010,	001011010.
O_1	O_2	O_3	O_4

One can easily verify or prove, as in Example 3.3, that these generalized necklaces satisfy the conditions of Construction 1 with $\delta = 2$, and hence give rise to a snake of length 56.

In the next, we restrict ourselves to the case $\delta = 2$ (snake-in-the-box codes). Under this restriction, we will extensively discuss a class of generalized necklaces that is obtained by taking $\pi = \sigma^2$ and applying π to words of even length. If we call the old notion of necklace 1-*necklace*, then this generalized necklace can be called 2-*necklace*. Two different words are said to be in the same 2-necklace, if one of them can be obtained from the other by applying π a finite number of times. Similarly, we may define the notion of *full* 2-necklaces as 2-necklaces with length $2n$ that have period n. Of course, one can equally well define 2-necklaces for words of odd length, but in that case the 2-necklace contains the same words as the corresponding 1-necklace. In case the word length is even, there are many full 2-necklaces that are not full 1-necklaces. We will show in Corollary 3.9 that the number of full 2-necklaces is twice the number of full 1-necklaces.

So, instead of using the permutation σ, by which any word of length $2n$

$$\mathbf{w} = (w_0, w_1, \ldots, w_{2n-2}, w_{2n-1})$$

is converted to

$$\sigma\mathbf{w} = (w_1, w_2, \ldots, w_{2n-1}, w_0),$$

we use the permutation π, by which the same word will be converted to

$$\pi\mathbf{w} = \sigma^2\mathbf{w} = (w_2, w_3, \ldots, w_0, w_1).$$

3.3. A More General Construction

The method of generating snakes by Construction 2 reduces to the one by Construction 1, if we apply the new notion of 2-necklace in the conditions of Construction 1. E.g. in [24, Section IV, Appendix II], the (11 + 1) different 1-necklaces of the snake with word length 8 constitute 24 different 2-necklaces. Similarly do the (33 +1) different 1-necklaces of the snake with word length 10, which give rise to 68 different 2-necklaces. In addition, the adapted Construction 1, which uses the 2-necklace notion, does work in the following construction.

Example 3.7.

The following eight words with length 6 represent eight different full 2-necklaces such that two consecutive 2-necklaces are uniquely at distance 1 with respect to 2,

000011, 000001, 001001, 101001, 101101, 111101, 111100, 011100.

Notice that with respect to 1-necklaces, these necklaces represent only six full 1-necklaces and the remaining two 1-necklaces are not full-periode. In the next Theorem 3.7 we shall see that by applying an adapted version of Construction 1 (we shall call it Construction 3) to these 2-necklaces, we get an almost optimal snake of length 24 in Q_6.

000011 000001 001001 101001 101101 111101 111100 011100
001100 000100 100100 100110 110110 110111 110011 110001
110000 010000 010010 011010 011011 011111 001111 000111

We remark that this snake (or an equivalent one, containing the same words in different order) can not be obtained by Construction 1 or 2 as formulated in [24]. This example illustrates furthermore that our approach is simpler than the constructions in [24], since there is no restriction put on the weights of the 2-necklaces.

We suggested already that one can even further generalize the notion of 2-necklace to q-necklace, where q is any positive integer. Indeed, we define a *q-necklace* $N = [\mathbf{w}]_q$ as the equivalence class in Q_n that contains $\mathbf{w}$ and is generated by the permutation $\pi = \sigma^q$, $q \geq 1$. In some cases, where no confusion occurs, e.g. when $q = 1$, we shall drop the subscript '$_q$' by simply writing $N = [\mathbf{w}]$.

The following theorem and its corollary suggest that q-necklaces with word length n are important only when $\gcd(n,q) > 1$. For $q = 2$ and n is even, we show that Construction 2 satisfies certain conditions. It is obvious that the word length n must be even. In fact, in the two examples presented in [24], the values of n are 8 and 10, respectively.

The following theorem leads to a conclusion (Corollary 3.6) that for every positive integer n and for every divisor q of n, any 1-necklace splits into $\gcd(n, q)$ distinct q-necklaces. First we provide the necessary and sufficient conditions for two words $\mathbf{w}$ and $\sigma^l\mathbf{w}$ belonging (or, not belonging) to the same q-necklace.

Theorem 3.5

Suppose $\mathbf{w} = (a_0, a_1, \ldots, a_{n-1})$ *is a word of length n that represents a full* 1*-necklace. Let q and l be two positive integers with* $0 < l < q < n$, *and let* $\rho := \gcd(n,q)$. *If* $M := [\mathbf{w}]_q$ *and* $N := [\sigma^l\mathbf{w}]_q$ *are two q-necklaces, then the following three statements are equivalent.*

a. for every integer k, $\sigma^{kq}(\sigma^l\mathbf{w}) \neq \mathbf{w}$;

b. for every integer k, there is an index i with $a_{i-kq-l} \neq a_i$;

c. for every integer k, $kq + l \neq 0 \bmod n$.

If the above equivalent statements are true, then so is the following statement

d. $\rho > 1$ *and l can not be multiples of* ρ.

Furthermore, M and N are two distinct q-necklaces, and if $q = \rho$, *then all four statements are equivalent. In particular, if* $q = \rho = 2$, *then n must be even and l must be odd.*

Proof. Let $\sigma^l\mathbf{w} = (b_0, b_1, \ldots, b_{n-1})$ and $\sigma^{kq}(\sigma^l\mathbf{w}) = (c_0, c_1, \ldots, c_{n-1})$. In terms of the components of $\sigma^{kq}(\sigma^l\mathbf{w})$ and $\mathbf{w}$, the statement (*a*) means that for every integer k, there exists an index i with $0 \leq i \leq n-1$ such that $c_i \neq a_i$, i.e.

$$c_i = b_{i-kq} = a_{i-kq-l} \neq a_i,$$

which is the statement (*b*). We have proven that (*a*) holds if and only if (*b*) holds. Recall that the period of a word in a full 1-necklace equals its length. Thus, $\mathbf{w}$ is of period n, and consequently, (*a*) is true if and only if for every integer k, $kq + l$ is not multiple of n, i.e. (*a*)

holds if and only if (*c*) holds.

Now notice that ρ is the generator of the cyclic additive subgroup of integers

$$\rho\mathbf{Z} = \{\ldots, -2\rho, -\rho, 0, \rho, 2\rho, \ldots\} = \{sn + tq \mid s, t \in \mathbf{Z}\} \tag{3.2}$$

of $\mathbf{Z}$. The RHS of the second equality in (3.2) states that equivalently, $\rho\mathbf{Z}$ is the group generated by the two integers n and q. Thus, for any integral multiple of ρ, say $r\rho$, there exist integers s and t such that $r\rho = sn + tq$. If we write $t = -k'$, then we may write this equality as

$$k'q + r\rho = sn. \tag{3.3}$$

If we apply the group homomorphism ϕ from $\mathbf{Z}$ onto $\mathbf{Z}_q = \phi(\mathbf{Z})$ based on congruence modulo q over the integers, the above subgroup is mapped onto the finite subgroup

$$\phi(\rho\,\mathbf{Z}) = \{0, \rho, 2\rho, \ldots, (q/\rho - 1)\rho\}$$

of $\mathbf{Z}_q$, and the element at the LHS of (3.3) is mapped onto the element $r\rho \in \phi(\rho\mathbf{Z}) \subseteq \mathbf{Z}_q$ with $0 \le r \le q/\rho - 1$. We conclude that any integral multiple of n, say sn, is mapped onto a multiple of ρ, say $r\rho$, which is an element of $\phi(\rho\mathbf{Z})$.

Now, suppose l is a multiple of ρ, say $l = r\rho \in \phi(\rho\mathbf{Z})$ with $0 \le r \le q/\rho - 1$. Then, there exist integers s and k' such that $k'q + r\rho = k'q + l = sn$. Consequently, (*c*) is false. It follows that if (*c*) is true, l can not be a multiple of ρ. This proves the second part of (*d*). Clearly, either $\rho > 1$ or $\rho = 1$. If l is not a multiple of ρ, then $l \notin \rho\mathbf{Z}$ and we have $\rho\mathbf{Z} \neq \mathbf{Z}$. Obviously, this is true if and only if $\rho > 1$.

Remember that M is a full q-necklace containing words of the form $\sigma^{kq}(\mathbf{w})$, whereas N is a full q-necklace containing words of the form $\sigma^{kq}(\sigma^{l}\mathbf{w})$. Therefore, if (*a*) is true, i.e. if for any integer k, $\sigma^{kq}(\sigma^{l}\mathbf{w}) \neq \mathbf{w}$, then the two full q-necklaces must be distinct. Conversely, if l is not a multiple of ρ, then for every integer k, $k\rho + l$ is not a multiple of ρ. Furthermore, if $q = \rho > 1$, which is a proper divisor of n, we must have that for every integer k, $kq + l$ is not a multiple of q. This implies for every integer k, that the integer $kq + l$ is not a multiple of n. We have just shown that if $q = \rho$ and (*d*) holds, then (*c*) holds, and hence in the case that $q = \rho > 1$, all four statements are equivalent.

Finally, suppose $\rho = q = 2$. Obviously, n is a multiple of $\gcd(n,q) = \rho = 2$. Thus n is even. Since we assumed that the four statements are true, l is not a multiple of $\rho = q = 2$. We conclude that l must be odd. □

The above theorem shows that for every positive $n > 2$, a full 1-necklace splits into $\rho = \gcd(n, q)$ distinct full q-necklaces. Briefly, one can say that the orbit of $\mathbf{w}$ under the action of σ splits into ρ smaller orbits under the actions of $\pi = \sigma^{q}$.

The second part of the theorem also suggests that if $\rho = \gcd(n,q) > 1$, then for any multiple of q, say kq, we have $\sigma^{kq}(\mathbf{w})$ and $\sigma^{k\rho}(\mathbf{w})$ are in the same q-necklace. So, it would be

computationally simpler to apply ρ-necklaces, rather than to apply q-necklaces with $q > \rho = \gcd(q, n)$.

Obviously, we can choose more than q possible values for l in Theorem 3.5. But, in case $\rho > 1$, the Theorem states that for every positive integer $l \in \mathbf{Z}$, we can find another positive integer $l' = \phi(l) < q$ such that $\sigma^{kq}(\sigma^{l}\mathbf{w})$ and $\sigma^{kq}(\sigma^{l'}\mathbf{w})$ are in the same q-necklace. For if $q \le l$, there would exist an integer l' such that $l = kq + l'$ with $0 \le l' < q$. Consequently, the words $\sigma^{l}\mathbf{w}$ and $\sigma^{l'}\mathbf{w}$ would represent the same q-necklace. Thus, we can assume w.lo.g. that $0 \le l < q$. The following corollary replaces this inequality by the stronger inequality $l < \rho$, i.e. the number of different q-necklaces is ρ.

Corollary 3.6

Let $\mathbf{w}$ *be a word of length n, let* $1 \le q < n$ *and let* $M = [\mathbf{w}]$ *be a full* 1*-necklace of binary words. If* $\rho = gcd(n,q)$ *and if the statements in Theorem* 3.5 *are true, then there are exactly* ρ *distinct full q-necklaces (or in particular, distinct* ρ*-necklaces*)

$$N_0 = [\mathbf{w}]_q, \quad N_1 = [\sigma\mathbf{w}]_q, \quad \ldots, \quad N_{\rho-1} = [\sigma^{\rho-1}\mathbf{w}]_q$$

such that $N_0 \cup N_1 \cup \ldots \cup N_{\rho-1} = M$. *In addition, these q-necklaces are of period* $p = n/\rho$ *and they are at distance at least* 2 *from each other.*

This corollary states that there are exactly ρ distinct values $l = 0, 1, \ldots, \rho - 1$. Therefore by relabelling or permuting bit-poisitions if neccessary, any q-necklace with $q > \rho$ can be replaced by a ρ-necklace. So in the sequel, we always assume that $\rho = \gcd(n, q) = q$.

Proof. We first claim that there will be at most ρ full q-necklaces. Let $N_l = [\sigma^{l}\mathbf{w}]_q$ be any q–necklace with $\rho \le l$. Thus, there are integers l'' and r such that $l = r\rho + l''$ and $0 \le l'' < \rho$. Now, by (3.3), there are also integers k' and s such that $k'q + r\rho = sn$. We have

$$\sigma^{k'q}(\sigma^{l}\mathbf{w}) = \sigma^{k'q}(\sigma^{r\rho+l''}\mathbf{w}) = \sigma^{l''}(\sigma^{k'q+r\rho}\mathbf{w}) = \sigma^{l''}(\sigma^{sn}\mathbf{w}) = \sigma^{l''}(\mathbf{w}),$$

since $\mathbf{w}$ is of length n. This proves that the full q-necklaces N_l and $N_{l''}$ with $0 \le l'' < \rho$ are identical. Therefore, all full q-necklaces can be represented by at most ρ full q-necklaces $N_l = [\sigma^{l}\mathbf{w}]_q$ with $0 \le l < \rho$.

In order to show that there will be no less than ρ full q-necklaces, it is sufficient to prove that for every integer l with $0 < l < \rho$, the two ρ-necklaces $N_0 = [\mathbf{w}]_q$ and $N_l = [\sigma^{l}\mathbf{w}]_q$ are distinct and at distance at least 2.

By hypothesis, statement (*a*) of Theorem 3.5 is true. This implies, obviously, that $\mathbf{w}$ is neither the zeroword nor the all-one word. Therefore with respect to σ^{q}, the words $\mathbf{w}$ and $\sigma^{l}\mathbf{w}$ represent different q-necklaces. The last part of the corollary is deduced from the fact that a full word and its shifts can only be at distance at least 2, or otherwise at distance 0. □

Now for the case $\rho > 1$, we are ready to adapt Construction 1. Recall that Theorem 3.5 and its corollary allow us to assume throughout the next pages and sections that $q = \rho$.

Theorem 3.7 (*Construction* 3)

Theorem 3.1 *is still valid for q-necklaces, $q \geq 1$, i.e. it is still true if the word 'necklace' is replaced by 'q-necklace'. In other words, suppose $t \geq \delta$, and let $N_0, N_1, \ldots, N_{t-1}$ be distinct, full-period q-necklaces of binary words of length pq such that for every i with $0 \leq i < t$, all indices taken modulo t, the following conditions hold:*

(i) *N_i and N_{i+r} are uniquely at distance r with respect to δ, for $0 \leq r < \delta$;*

(ii) *N_i and N_{i+r} are at distance $\geq \delta$, for $\delta \leq r \leq t - \delta$.*

It follows that there exist words $\mathbf{w}_i, \mathbf{w}_{i+1} \in N_i$ with $d(\mathbf{w}_i, \mathbf{w}_{i+1}) = 1$, $0 \leq i < t$, and an integer l, $0 \leq l < n$, with $d(\mathbf{w}_{t-1}, \sigma^l \mathbf{w}_0) = 1$. Furthermore, there is an integer m such that the list C obtained by concatenation of the rows

$$\begin{array}{cccc} \mathbf{w}_0, & \mathbf{w}_1, & \mathbf{w}_2, & \ldots, \mathbf{w}_{t-1} \\ \sigma^{lq}\mathbf{w}_0, & \sigma^{l}\mathbf{w}_1, & \sigma^{l}\mathbf{w}_2, & \ldots, \sigma^{l}\mathbf{w}_{t-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma^{(m-1)l}\mathbf{w}_0, & \sigma^{(m-1)l}\mathbf{w}_1, & \sigma^{(m-1)l}\mathbf{w}_2, & \ldots, \sigma^{(m-1)l}\mathbf{w}_{t-1} \end{array}$$

is a spread-δ circuit with mt words of length $n = pq$. Here, the integer m is defined as

$$m := \frac{n}{\gcd(l, n)} . \tag{3.4}$$

Proof. The proof is similar to the proof given in [24, Theorem 1] for $n = pq$. □

Now we show that method in Construction 2 does generate a snake. Recall that this method starts from t distinct, full-period 1-necklaces. More precisely, we show that applying Construction 2 with initially $2(t + 1)$ words in $t + 2$ distinct, full-period 1-necklaces (including $[\mathbf{v}_0]$ and $[\mathbf{v}_1]$) is equivalent with applying Construction 3. In particular, each of the t 1-necklaces different from $[\mathbf{v}_0]$ and from $[\mathbf{v}_1]$ has to be split into two distinct, full-period 2-necklaces of equal size, and such that the roles of the three integers l, t and m (expressed by (3.4)) in the Construction 3 above, are played by the integers $l + l'$, $2(t + 1)$ and m' (expressed by (3.7) below), respectively.

Theorem 3.8

Let $t \geq 2$, and assume that

$$M_0 = [\mathbf{w}_0], M_1 = [\mathbf{w}_1], \ldots, M_{t-1} = [\mathbf{w}_{t-1}] \tag{3.5}$$

are distinct, full-period 1-necklaces with word length $n = 2p$ that satisfy all the conditions in Construction 2. Then the weight-$(n - 1)$ and the weight-1 words $\mathbf{v}_0$ and $\mathbf{v}_1$, respectively, describe in Construction 2 do exist. Moreover, both integers l and l' are odd and the list of $2(t + 1)$ full 2-necklaces

$$N_0 = [\mathbf{u}_0]_2, N_1 = [\mathbf{u}_1]_2, \ldots, N_{t-1} = [\mathbf{u}_{t-1}]_2, N_t = [\mathbf{u}_t]_2, \tag{3.6}$$
$$N_{t+1} = [\mathbf{u}_{t+1}]_2, N_{t+3} = [\mathbf{u}_{t+2}]_2, \ldots, N_{2t} = [\mathbf{u}_{2t}]_2, N_{2t+1} = [\mathbf{u}_{2t+1}]_2$$

satisfy the conditions for Construction 3 *such that the integers l, t and m, as described in Construction* 3*, are replaced by the integers* $l + l'$, $2(t + 1)$ *and*

$$m' := \frac{p}{\gcd(\frac{l+l'}{2}, p)}, \tag{3.7}$$

respectively.

We remark that for every t satisfying $0 \le i \le t - 1$, the same binary word $\mathbf{w}_i = \mathbf{u}_i$ represents two different kinds of necklaces, a 1-necklace $M_i = [\mathbf{w}_i]$ and a 2-necklace $N_i = [\mathbf{w}_i]_2$.

Proof. In the seheme of Theorem 3.3 (Construction 2), the first t full 1-necklaces $M_0 = [\mathbf{w}_0]$, $M_1 = [\mathbf{w}_1]$, …, $M_{t-1} = [\mathbf{w}_{t-1}]$ determine a list of t distinct, full-period 2-necklaces $N_0 = [\mathbf{w}_0]_2$, $N_1 = [\mathbf{w}_1]_2$, …, $N_{t-1} = [\mathbf{w}_{t-1}]_2$. If $\mathbf{u}_{t+1} := \sigma^l \mathbf{w}_{t-1}$, $\mathbf{u}_{t+2} := \sigma^l \mathbf{w}_{t-2}$, …, $\mathbf{u}_{2t} := \sigma^l \mathbf{w}_0$, then the following t 1-necklaces $M_{t+1} = [\mathbf{u}_{t+1}]$, $M_{t+2} = [\mathbf{u}_{t+2}]$, …, $M_{2t} = [\mathbf{u}_{2t}]$ also determine a list of t distinct, full-period 2-necklaces $N_{t+1} = [\mathbf{u}_{t+1}]_2$, $N_{t+2} = [\mathbf{u}_{t+2}]_2$, …, $N_{2t} = [\mathbf{u}_{2t}]_2$.

From the last part of Theorem 3.5, k is odd if and only if the two 2-necklaces $[\sigma^k \mathbf{w}_i]_2$ and $[\mathbf{w}_i]_2$ are distinct. Therefore, for an arbitrary odd integer $k < n/2$ (= p, the period of a full-period 2-necklace), each of the t 2-necklaces $N_i = [\mathbf{w}_i]_2$, $0 \le i \le t - 1$, cannot be the same with any of the t 2-necklaces $N_{2t-i+1} = [\sigma^k \mathbf{w}_i]_2$, $0 \le i \le t - 1$.

Since $\mathbf{u}_{t-1}$ is a weight-$(n - 2)$ word, there is a weight-$(n - 1)$ 2-necklace N_t and a word $\mathbf{v}_0 \in N_t$ such that $d(N_t, N_{t+1}) = d(\mathbf{u}_{t-1}, \mathbf{v}_0) = 1$. Since $\mathbf{v}_0$ is a weight-$(n - 1)$ word, $\mathbf{u}_{t-1}$ is not the only weight-$(n - 2)$ word in 1-necklace $M_{t-1} = [\mathbf{u}_{t-1}]$ that is at distance 1 with $\mathbf{v}_0$, i.e. with respect to 1-necklaces, $d(M_{t-1}, [\mathbf{v}_0]) = 1$ *not* uniquely. Therefore, there exists nonzero integer $l < n/2$ such that $d(\sigma^l \mathbf{u}_{t-1}, \mathbf{v}_0) = 1$.

However, with respect to 2-necklaces, $d(N_{t-1}, N_t) = 1$ uniquely (with respect to 2). This fact together with the resulting equality $d(\mathbf{u}_{t-1}, \mathbf{v}_0) = d(\sigma^0 \mathbf{u}_{t-1}, \mathbf{v}_0) = 1 = d(\sigma^l \mathbf{u}_{t-1}, \mathbf{v}_0)$ imply that the two words $\mathbf{u}_{t-1} = \sigma^0 \mathbf{u}_{t-1}$ and $\sigma^l \mathbf{u}_{t-1}$ belong to different 2-necklaces. We conclude that l is odd. Similarly, since $\mathbf{u}_0$ is a weight-2 word, there exists an odd integer $l' < n/2$, a weight-1 2-necklace N_{2t+1} and a word $\mathbf{v}_1 \in N_{2t+1}$ such that $d(N_0, N_{2t+1}) = d(\mathbf{v}_1, \sigma^{l'} \mathbf{u}_0) = 1 = d(\sigma^{l+l'} \mathbf{u}_0, \mathbf{v}_1)$. So, if $\mathbf{u}_{2t+1} \in N_{2t+1}$ such that $\mathbf{v}_1 = \sigma^l \mathbf{u}_{2t+1}$, we can apply Construction 3 to generate a snake of length-n words started by the $2(t + 1)$ words

$$\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{t-2}, \mathbf{u}_{t-1}, \mathbf{u}_t, \tag{3.8}$$
$$\mathbf{u}_{t+1} = \sigma^l \mathbf{u}_{t-1}, \ \mathbf{u}_{t+2} = \sigma^l \mathbf{u}_{t-2}, \ \ldots, \ \mathbf{u}_{2t-1} = \sigma^l \mathbf{u}_1, \ \ \mathbf{u}_{2t} = \sigma^l \mathbf{u}_0, \ \sigma^l \mathbf{u}_{2t+1},$$

representing the $2(t + 1)$ full-period 2-necklaces of (3.6) obtained from the t 1-necklaces M_i. Next, by using these $2(t + 1)$ words, we show that these $2(t + 1)$ words play the role of the first

row of t words in the scheme stated in Theorem 3.7 (Construction 3).

Observe that this row of $2(t + 1)$ consists of two different groups of $t + 1$ consecutive words. The first group consists of $t + 1$ words belonging to the 2-necklaces in the first row of (3.6) and this group is followed by the second group consisting of $t + 1$ consecutive words belonging to the 2-necklaces in the second row of (3.6). By applying shifting operator $\sigma^{l+l'}$, we obtain the following second row of the scheme

$$\sigma^{l+l'}\mathbf{u}_0, \sigma^{l+l'}\mathbf{u}_1, \ldots, \sigma^{l+l'}\mathbf{u}_{t-1}, \sigma^{l+l'}\mathbf{u}_t,$$
$$\sigma^{2l+l'}\mathbf{u}_{t-1}, \sigma^{2l+l'}\mathbf{u}_{t-2}, \ldots, \sigma^{2l+l'}\mathbf{u}_0, \sigma^{2l+l'}\mathbf{u}_{2t+1}.$$

If we apply the same shifting operator $\sigma^{l+l'}$ to all these words, we obtain another set of $2(t + 1)$ words. If we continue this procedure and choose *positive* odd integers l and l', then we finally obtain a set of $2(t + 1)$ words

$$\sigma^{(m'-1)(l+l')}\mathbf{u}_0, \sigma^{(m'-1)(l+l')}\mathbf{u}_1, \ldots, \sigma^{(m'-1)(l+l')}\mathbf{u}_{t-1}, \sigma^{(m'-1)(l+l')}\mathbf{u}_t,$$
$$\sigma^{(m'-1)(l+l')+l}\mathbf{u}_{t-1}, \sigma^{(m'-1)(l+l')+l}\mathbf{u}_{t-2}, \ldots, \sigma^{(m'-1)(l+l')+l}\mathbf{u}_0, \sigma^{(m'-1)(l+l')+l}\mathbf{u}_{2t+1}.$$

where (cf. Construction 3) $0 \le l, l' < n/2$ and

$$m' := \frac{n}{\gcd(l + l', n)}.$$

From the fact that $n = 2p$ and $l + l'$ is even, (3.7) follows.

Since l and l' are odd, $l + l'$ is even and the positions of $2(t + 1)$ consecutive 2-necklaces in (3.6) are not affected by the shifting operator $\sigma^{l+l'}$(which only permutes the $n/2$ words in each of those $2(t + 1)$ 2-necklaces). In particular, the last word obtained by applying the shifting operator $\sigma^{l+l'}$ $m' - 1$ times is the weight-1 word $\sigma^{(m'-1)(l+l')+l}\mathbf{u}_{2t+1}$. Since we assume Theorem 3.7 (Construction 3) is true, the next word of the weight-1 $\sigma^{(m'-1)(l+l')+l}\mathbf{u}_{2t+1}$ must be the same as the weight-2 word belonging to the first group of $t + 1$ words (3.8), i.e. $\sigma^{m'(l+l')}\mathbf{u}_0 = \mathbf{u}_0$.

We have just proved that from the given list of t distinct, full-period 1-necklaces stated in Construction 2, we obtain the list (3.6) of $2(t + 1)$ distinct, full-period 2-necklaces that satisfy the conditions of Construction 3. □

Notice that in the case $q = 2$, as is the case in Theorem 3.8 above, we actually have $m' = n/2$, which is the period of full-period 2-necklaces N_i, $0 \le i \le 2t + 1$, discussed in the proof of the theorem. Next, we state a more specific result under the assumption $q = 2 = \rho$.

Corollary 3.9

If in Construction 2 the word length n is even, then the integers l and l' are odd and the construction produces a snake of length 2m(t + 1)

Example 3.8

The snake obtained by Paterson and Tuliani in [24] is currently known as the longest, and is very likely the maximal, snake in Q_8. The first eleven ($t = 11$) words in the left column correspond to eleven full 1-necklaces. The code is generated by applying Construction 2. Since the integers $l = \pm 1$ and $l' = \pm 3$ are odd, Corollary 3.9 guarantees that there are $2(t+1) = 24$ distinct 2-necklaces. Because any word of a 1-necklace is a word of a 2-necklace, the previous eleven 1-necklaces provide us with almost half of the total number of the possible 2-necklaces. It can be shown that the weight of the 12-th necklace, containing $\mathbf{v}_0$, must be greater than the weight of each these eleven 2-necklaces. So, there is only one choice left for the word $\mathbf{v}_0$.

The remaining 2-necklaces, which are represented by the words in the second column below, are also 1-necklaces satisfying the conditions of Construction 2, but in reversed order. Moreover, the weight of the 24-th 2-necklace containing $\mathbf{v}_1$ must be less than the weight of each of these last eleven 2-necklaces. Applying Construction 3, we obtain the same snake as in [24].

1. 0 0 0 0 0 0 0 1 1;	13. 0 1 1 1 1 0 1 1;
2. 0 0 0 0 1 0 1 1;	14. 0 1 1 1 1 0 0 1;
3. 0 0 0 1 1 0 1 1;	15. 0 0 1 1 1 0 0 1;
4. 0 0 0 1 1 1 1 1;	16. 0 0 1 0 1 0 0 1;
5. 0 0 0 1 1 1 0 1;	17. 1 0 1 0 1 0 0 1;
6. 0 0 0 1 0 1 0 1;	18. 1 0 1 0 1 0 0 0;
7. 0 0 1 1 0 1 0 1;	19. 1 1 1 0 1 0 0 0;
8. 0 0 1 0 0 1 0 1;	20. 1 1 1 1 1 0 0 0;
9. 0 0 1 0 0 1 1 1;	21. 1 1 0 1 1 0 0 0;
10. 0 0 1 0 1 1 1 1;	22. 0 1 0 1 1 0 0 0;
11. 0 1 1 0 1 1 1 1;	23. 0 0 0 1 1 0 0 0;
12. 0 1 1 1 1 1 1 1;	24. 0 0 0 0 1 0 0 0.

With respect to Construction 2, the two left shifts are those over $l = 3$ and over $l' = 7$ positions whereas with respect to the Construction 3, the left shift is the one over $l + l' \bmod n = 10 \bmod 8 = 2$ positions. The resulting code is a snake of length 96. The code can also be generated using the following transition sequence

4, 3, 5, 6, 4, 2, 3, 6, 4, 1, 3, 5, 6, 1, 3, 0, 7, 1, 3, 2, 0, 1, 3, 5,
2, 1, 3, 4, 2, 0, 1, 4, 2, 7, 1, 3, 4, 7, 1, 6, 5, 7, 1 ,0, 6, 7, 1, 3,
0, 7, 1, 2, 0, 6, 7, 2, 0, 5, 7, 1, 2, 5, 7, 4, 3, 5, 7, 6, 4, 5, 7, 1,
6, 5, 7, 0, 6, 4, 5, 0, 6, 3, 5, 7, 0, 3, 5, 2, 1, 3, 5, 4, 2, 3, 5, 7.

When applying the Gray map

$$00 \mapsto 0, \quad 10 \mapsto 1, \quad 11 \mapsto 2, \quad 01 \mapsto 3$$

to the above binary code, it is obvious that the resulting quarternary code is a '*single-track* 2-*circuit*', i.e. the four columns (or 'tracks') can be obtained from each other by shifting (cf. [12,13]).

1. 0 0 0 2;	25. 0 0 2 0;	49. 0 2 0 0;	73. 2 0 0 0;
2. 0 0 1 2;	26. 0 1 2 0;	50. 1 2 0 0;	74. 2 0 0 1;
3. 0 3 1 2;	27. 3 1 2 0;	51. 1 2 0 3;	75. 2 0 3 1;
4. 0 3 2 2;	28. 3 2 2 0;	52. 2 2 0 3;	76. 2 0 3 2;
5. 0 3 2 3;	29. 3 2 3 0;	53. 2 3 0 3;	77. 3 0 3 2;
6. 0 3 3 3;	30. 3 3 3 0;	54. 3 3 0 3;	78. 3 0 3 3;
7. 0 2 3 3;	31. 2 3 3 0;	55. 3 3 0 2;	79. 3 0 2 3;
8. 0 1 3 3;	32. 1 3 3 0;	56. 3 3 0 1;	80. 3 0 1 3;
9. 0 1 3 2;	33. 1 3 2 0;	57. 3 2 0 1;	81. 2 0 1 3;
10. 0 1 2 2;	34. 1 2 2 0;	58. 2 2 0 1;	82. 2 0 1 2;
11. 3 1 2 2;	35. 1 2 2 3;	59. 2 2 3 1;	83. 2 3 1 2;
12. 3 2 2 2;	36. 2 2 2 3;	60. 2 2 3 2;	84. 2 3 2 2;
13. 3 2 1 2;	37. 2 1 2 3;	61. 1 2 3 2;	85. 2 3 2 1;
14. 3 2 1 3;	38. 2 1 3 3;	62. 1 3 3 2;	86. 3 3 2 1;
15. 0 2 1 3;	39. 2 1 3 0;	63. 1 3 0 2;	87. 3 0 2 1;
16. 0 1 1 3;	40. 1 1 3 0;	64. 1 3 0 1;	88. 3 0 1 1;
17. 1 1 1 3;	41. 1 1 3 1;	65. 1 3 1 1;	89. 3 1 1 1;
18. 1 1 1 0;	42. 1 1 0 1;	66. 1 0 1 1;	90. 0 1 1 1;
19. 2 1 1 0;	43. 1 1 0 2;	67. 1 0 2 1;	91. 0 2 1 1;
20. 2 2 1 0;	44. 2 1 0 2;	68. 1 0 2 2;	92. 0 2 2 1;
21. 2 3 1 0;	45. 3 1 0 2;	69. 1 0 2 3;	93. 0 2 3 1;
22. 3 3 1 0;	46. 3 1 0 3;	70. 1 0 3 3;	94. 0 3 3 1;
23. 0 3 1 0;	47. 3 1 0 0;	71. 1 0 0 3;	95. 0 0 3 1;
24. 0 0 1 0;	48. 0 1 0 0;	72. 1 0 0 0;	96. 0 0 0 1.

Example 3.9

Similarly, as in Example 3.8, the first thirty three words in the following sequence of length-10 words represent thirty-three full 1-necklaces given as an example in [24]. The thirty-three 1-necklaces satisfy the conditions of Construction 2. The 34-th and the 68-th words are the 'heaviest' weight-9 word $\mathbf{v}_0$ and the 'lightest' weight-1 word $\mathbf{v}_1$, respectively, mentioned in the proof of Theorem 3.8.

1. 0 0 0 0 0 0 0 0 1 1;	35. 1 1 1 1 0 0 1 1 1 1;
2. 0 0 0 0 0 0 0 1 0 1;	36. 1 1 1 1 0 0 1 0 1 1;
3. 0 0 0 0 0 0 0 1 1 1;	37. 1 1 1 1 0 0 1 0 0 1;
4. 0 0 0 0 1 0 1 1 1 1;	38. 0 1 1 1 0 0 1 0 0 1;
5. 0 0 0 0 1 0 0 1 1 1;	39. 0 1 1 1 0 0 1 0 0 0;
6. 0 0 0 0 1 0 0 1 0 1;	40. 0 1 0 1 0 0 1 0 0 0;
7. 0 0 0 0 1 1 0 1 0 1;	41. 1 1 0 1 0 0 1 0 0 0;
8. 0 0 1 0 1 1 0 1 0 1;	42. 1 1 0 1 0 0 1 0 1 0;
9. 0 0 1 1 1 1 0 1 0 1;	43. 1 1 0 1 0 0 1 1 1 0;
10. 0 1 1 1 1 1 0 1 0 1;	44. 1 1 0 1 0 1 1 1 1 0;
11. 0 1 1 1 1 1 0 1 0 0;	45. 1 1 0 0 0 1 1 1 1 0;
12. 0 1 0 1 1 1 0 1 0 0;	46. 1 1 0 0 0 1 0 1 1 0;
13. 0 1 0 0 1 1 0 1 0 0;	47. 1 1 0 0 1 1 0 1 1 0;
14. 0 1 0 0 0 1 0 1 0 0;	48. 1 1 0 0 1 1 0 0 1 0;
15. 0 1 0 0 0 1 1 1 0 0;	49. 1 1 0 0 1 1 1 0 1 0;
16. 0 1 0 1 0 1 1 1 0 0;	50. 1 1 1 0 1 1 1 0 1 0;
17. 0 1 1 1 0 1 1 1 0 0;	51. 1 1 1 0 1 1 1 0 0 0;
18. 0 1 1 1 0 1 1 1 0 1;	52. 1 0 1 0 1 1 1 0 0 0;

19. 0 1 1 0 0 1 1 1 0 1;	53. 1 0 0 0 1 1 1 0 0 0;
20. 0 1 1 0 0 1 1 0 0 1;	54. 1 0 0 0 1 0 1 0 0 0;
21. 0 1 1 0 0 1 1 0 1 1;	55. 1 0 0 1 1 0 1 0 0 0;
22. 0 1 1 0 0 0 1 0 1 1;	56. 1 0 1 1 1 0 1 0 0 0;
23. 0 1 1 0 0 0 1 1 1 1;	57. 1 1 1 1 1 0 1 0 0 0;
24. 0 1 1 0 1 0 1 1 1 1;	58. 1 1 1 1 1 0 1 0 1 0;
25. 0 1 1 0 1 0 0 1 1 1;	59. 0 1 1 1 1 0 1 0 1 0;
26. 0 1 1 0 1 0 0 1 0 1;	60. 0 1 0 1 1 0 1 0 1 0;
27. 0 1 1 0 1 0 0 1 0 0;	61. 0 0 0 1 1 0 1 0 1 0;
28. 0 0 1 0 1 0 0 1 0 0;	62. 0 0 0 1 0 0 1 0 1 0;
29. 0 0 1 1 1 0 0 1 0 0;	63. 0 0 0 1 0 0 1 1 1 0;
30. 1 0 1 1 1 0 0 1 0 0;	64. 0 0 0 1 0 1 1 1 1 0;
31. 1 1 1 1 1 0 0 1 0 0;	65. 0 0 0 0 0 1 1 1 1 0;
32. 1 1 1 1 1 0 0 1 0 1;	66. 0 0 0 0 0 1 0 1 1 0;
33. 1 1 1 1 1 0 0 1 1 1;	67. 0 0 0 0 0 0 0 1 1 0;
34. 1 1 1 1 1 0 1 1 1 1;	68. 0 0 0 0 0 0 0 1 0 0.

With respect to Construction 2, both the two left shifts are those over $l = 1 = l'$ position. So, with respect to the Construction 3, the left shift is the one over $l + l' \bmod 10 = 2$ positions.

3.4. The Structures of 2-Necklaces in Q_{2p}, $p > 2$ prime.

If p is a prime number, then the notion of q-necklace in Q_p is meaningless, e.g. if an a-necklace and a b-necklace contain the same word, then the two necklaces coincide. So, we simply speak of 'a necklace' for any q-necklace in Q_p with $1 < q < p$. Moreover, every necklace is full, i.e. its period is p.

For every integer i satisfying $1 \le i \le p - 1$, the number of distinct weight-i words is

$$\binom{p}{i} = \frac{p(p-1)\ldots(p-i+1)}{i(i-1)\ldots(2)(1)}.$$

Since every necklace in Q_p is of full period p,

$$N_p = \sum_{i=1}^{p-1} \binom{p}{i} / p = (2^p - 2)/p \tag{3.9}$$

is the number of different necklaces in Q_p.

Example 3.10

For $p = 5$, we have $N_5 = (5 + 10 + 10 + 5)/5 = 6$ distinct 2-necklaces. As representatives for these 2-necklaces, we take

$$\mathbf{p} = 00001, \quad \mathbf{q} = 10001, \quad \mathbf{r} = 10101, \quad \mathbf{s} = 10100, \quad \mathbf{t} = 11100, \quad \mathbf{u} = 11110.$$

Let $L = 2p$. The following notation assigns every (period-p) necklace U in Q_p to a particular (indexed) word

$$\mathbf{u}_0 = u_0 u_1 u_2 \ldots u_{p-1} \in U.$$

We introduce the following additional notation imposed on all words $\sigma^i(\mathbf{u}_0)$, $1 \le i \le p - 1$, of U

$$\mathbf{u}_i = u_i u_{i+1} \ldots u_{p-1} u_0 u_1 \ldots u_{i-1} = \sigma^i(\mathbf{u}_0) \in U.$$

Moreover, if we write $\mathbf{0}_n$ and $\mathbf{1}_n$ for the all-0 and all-1 n-tuples in Q_n and if

$$\mathbf{v}_0 = v_0 v_1 v_2 \ldots v_{p-1}$$

is a fixed word in Q_p, then we define length-$2p$ words $\mathbf{u}_i\mathbf{v}_j$ in Q_{2p} as follows

$$\mathbf{u}_i\mathbf{v}_j = u_i\, v_j\, u_{i+1}\, v_{j+1} \ldots u_{i-1}\, v_{j-1}, \tag{3.10}$$

where $\mathbf{u}_i \neq \mathbf{0}_p, \mathbf{1}_p$ or $\mathbf{v}_j \neq \mathbf{0}_p, \mathbf{1}_p$. Obviously, with respect to the notion of 2-necklace, the words $\mathbf{u}_i\mathbf{v}_j$ are always of full period-p.

By using all pairs of two words $\mathbf{u}_i$ and $\mathbf{v}_j$ with $\mathbf{u}_i \neq \mathbf{0}_p, \mathbf{1}_p$ or $\mathbf{v}_j \neq \mathbf{0}_p, \mathbf{1}_p$ to obtain distinct, period-p length-$2p$ words $\mathbf{u}_i\,\mathbf{v}_j$, we are practically able to list all distinct 2-necklaces in Q_{2p}. In fact, for every pair of nonzero words $\mathbf{u}_0$ and $\mathbf{v}_0$ in Q_p, and for every (period-p) 2-necklace N in Q_{2p}, we observe that

$$\mathbf{u}_i\mathbf{v}_j \in N,$$

if and only if

$$\mathbf{u}_0\mathbf{v}_{j-i \bmod p} \in N,$$

or if and only if

$$\mathbf{u}_{i-j \bmod p}\mathbf{v}_0 \in N.$$

Therefore, by fixing either the index i of $\mathbf{u}_i$, e.g. by fixing $i = 0$, or the index j of $\mathbf{v}_j$, shifting the other index will generate additional words of distinct, period-p 2-necklaces.

Example 3.11

In case $L = 10 = 2p$, there are six distinct necklaces in Q_5 (cf. Example 3.10). Two of the necklaces are represented by $\mathbf{p}_0 = 00001$ and $\mathbf{q}_0 = 10001$. The following is a list of some words from distinct, period-5 2-necklaces in Q_{10} that are obtained by shifting the indices of the two words $\mathbf{p}_0$ and $\mathbf{q}_0$.

$\mathbf{p}_0\mathbf{p}_0 = 0000000011$; $\mathbf{p}_0\mathbf{p}_1 = 0000000110$;
$\mathbf{p}_2\mathbf{q}_1 = 0000100101$; $\mathbf{p}_1\mathbf{q}_0 = 0100001001$;
$\mathbf{p}_0\mathbf{q}_2 = 0000010110$; $\mathbf{q}_1\mathbf{p}_0 = 0000001011$;
$\mathbf{p}_0\mathbf{q}_1 = 0000000111$; $\mathbf{p}_1\mathbf{q}_2 = 0000011100$.

This list shows that $\mathbf{p}_0\mathbf{q}_1 \neq \mathbf{p}_1\mathbf{q}_2$, but $[\mathbf{p}_0\mathbf{q}_1]_2 = [\mathbf{p}_1\mathbf{q}_2]_2$ and $\mathbf{p}_2\mathbf{q}_1 \neq \mathbf{p}_1\mathbf{q}_0$ but $[\mathbf{p}_2\mathbf{q}_1]_2 = [\mathbf{p}_1\mathbf{q}_0]_2$. It also shows that although $\mathbf{p}_0\mathbf{q}_2$ and $\mathbf{q}_1\mathbf{p}_0$ are in the same 1-necklace of Q_{10}, both words are from different 2-necklaces. The same situation also happens between the two length-$2p$ words $\mathbf{p}_0\mathbf{p}_0$ and $\mathbf{p}_0\mathbf{p}_1$.

In general, unless $\mathbf{u}_i = \mathbf{v}_j$ for some i and j, for every pair of nonzero words $\mathbf{u}_i$ and $\mathbf{v}_j$ in Q_p with $1 \leq i, j < p$, the 2-necklace $[\mathbf{u}_i\mathbf{v}_j]_2$ in Q_{2p} cannot contain the word $\mathbf{v}_j\mathbf{u}_i$, i.e. the 2-necklace $[\mathbf{u}_i\,\mathbf{v}_j]_2$ is different from the 2-necklace $[\mathbf{v}_j\,\mathbf{u}_i]_2$, despite the fact that the two length-$2p$ words $\mathbf{u}_i\,\mathbf{v}_j$ and $\mathbf{v}_j\,\mathbf{u}_i$ may belong to the same 1-necklace.

Since every 1-necklace in Q_p as well as every 2-necklace in Q_{2p} is of period p, and since there are N_p distinct 1-necklaces in Q_p, we conclude that a pair of two nonzero words $\mathbf{u}_0$, $\mathbf{v}_0$ taken from *distinct* necklaces in Q_p determine $pN_p(N_p - 1)/2$ distinct period-p 2-necklaces in Q_{2p} and these necklaces contain length-$2p$ words of the form $\mathbf{u}_i\mathbf{v}_j$, $1 \leq i, j < p$. Similarly, the

same pair of words determine $pN_p(N_p - 1)/2$ distinct period-p 2-necklaces containing length-$2p$ words of the form $\mathbf{v}_j\,\mathbf{u}_i$, $1 \le i, j < p$.

Now if $p > 2$, then p is odd and this implies that the number of terms in the sum for N_p in (3.9) is even. In fact, half of the number of terms are expressing the number of 2-necklaces containing all words of the form $\mathbf{u}_i\mathbf{v}_j$ whereas the other half of the terms are expressing the number of 2-necklaces containing all words of the form $\mathbf{u}^c{}_i\mathbf{v}^c{}_j$, where $\mathbf{u}_i$ and $\mathbf{v}_j$ are two words from two distinct necklaces in Q_p and $\mathbf{u}^c = \mathbf{1}_p + \mathbf{u}$. We conclude that there are $pN_p(N_p - 1)$ distinct, period-p 2-necklaces in Q_{2p} that contain words of the form $\mathbf{u}_i\mathbf{v}_j$, where $\mathbf{u}_i$ and $\mathbf{v}_j$ are from two distinct necklaces in Q_p.

Next, from all words of the form $\mathbf{u}_0\mathbf{u}_j$ (or $\mathbf{u}_i\mathbf{u}_0$), $1 \le i < p$, we obtain pN_p more distinct period-p 2-necklaces. Finally, all N_p necklaces containing $\mathbf{u}_0 \ne \mathbf{0}_p$, $\mathbf{1}_p$, can be used to produce $4N_p$ more distinct, period-p 2-necklaces containing the words of the form $\mathbf{0}_p\mathbf{u}_i$, $\mathbf{1}_p\mathbf{u}_i$, $\mathbf{u}_i\mathbf{0}_p$ or $\mathbf{u}_i\mathbf{1}_p$. Totally, there are

$$pN_p(N_p - 1) + pN_p + 4N_p = N_p(p(N_p - 1) + p + 4).$$

distinct 2-necklaces of period-p.

On the other hand, among the 2^{2p} words in Q_{2p}, there are exactly 4 words not contained in the period-p 2-necklaces already obtained above. These are the two $\mathbf{0}_{2p}$, $\mathbf{1}_{2p}$ and the two period-2, weight-p words 0101…01 and 1010…10. Therefore, the number of distinct 2-necklaces equals $(2^{2p} - 4)/p$.

We combine the above two results in the following theorem.

Theorem 3.10

If $p > 2$ is prime, then there are

$$N_p(p(N_p - 1) + p + 4) = (2^{2p} - 4)/p \tag{3.11}$$

distinct, period-p 2-necklaces in Q_{2p}, where N_p is expressed by (3.9).

Example 3.12

Let $L = 6 = 2p$. Here, $\mathbf{u}_0 = 001$ and $\mathbf{v}_0 = 011$ represent all the $N_3 = 2$ distinct 1-necklaces in Q_3.

These two distinct words $\mathbf{u}_0$ and $\mathbf{v}_0$ in $Q_3 \setminus \{\mathbf{0}_3, \mathbf{1}_3\}$ determine

$$3N_3(N_3 - 1) = 6$$

period-3 2-necklaces in Q_6 that contain the following words

$$\mathbf{u}_0\mathbf{v}_0 = 000111, \quad \mathbf{u}_0\mathbf{v}_1 = 010110, \quad \mathbf{u}_0\mathbf{v}_2 = 010011,$$
$$\mathbf{v}_0\mathbf{u}_0 = 001011, \quad \mathbf{v}_1\mathbf{u}_0 = 101001, \quad \mathbf{v}_2\mathbf{u}_0 = 100011.$$

We get $3N_3 = 6$ more distinct 2-necklaces represented by

$$\mathbf{u}_0\mathbf{u}_0 = 000011,\ \mathbf{u}_0\mathbf{u}_1 = 000110,\ \mathbf{u}_0\mathbf{u}_2 = 010010$$

and

$$\mathbf{v}_0\mathbf{v}_0 = 001111,\ \mathbf{v}_0\mathbf{v}_1 = 011110,\ \mathbf{v}_0\mathbf{v}_2 = 011011.$$

Finally, we get $4N_3 = 8$ more distinct 2-necklaces from

$$\mathbf{0}_3\mathbf{u}_0 = 000001, \qquad \mathbf{1}_3\mathbf{u}_0 = 101011,$$
$$\mathbf{0}_3\mathbf{v}_0 = 000101, \qquad \mathbf{1}_3\mathbf{v}_0 = 101111,$$
$$\mathbf{u}_0\mathbf{0}_3 = 000010, \qquad \mathbf{u}_0\mathbf{1}_3 = 010111,$$
$$\mathbf{v}_0\mathbf{0}_3 = 001010, \qquad \mathbf{v}_0\mathbf{1}_3 = 011111.$$

Totally, there are $3 \cdot 20 = 60$ words in the 20 2-necklaces above. The left $2^6 - 60 = 4$ words are not contained by any of these 2-necklaces. These four words are $\mathbf{0}_6$, $\mathbf{1}_6$, 010101 and 101010.

The even number of terms in (3.9), as mentioned before in the proof of Theorem 3.10, gives a clue why in [24], the number of (full) initial 2-necklaces used when constructing long snakes by applying the necklace approach, is always even. In fact, these initial 2-necklaces always split into two groups of necklaces of the same size. These two groups contain necklaces from two disjoint families $\mathscr{N}_1$ and $\mathscr{N}_2$, respectively, that contain $(2^{2p-1} - 2)/p$ period-p 2-necklaces such that any 2-necklace $N_1 \in \mathscr{N}_1$ containing weight-w words is the complement of a 2-necklace $N_2 \in \mathscr{N}_2$ containing weight-$(2p - w)$ words, $0 < w < 2p$.

4. Snakes Based on a Linear Code

In this chapter, we shall discuss a general method to construct circuit codes which was used already by van Zanten and Lukito to construct $<m,n>$-codes (cf. [39]) and snake-in-the-box codes (cf. [19, 41]). Before coming to the details of our construction in Section 4.2, we first present a global description and the basic ideas underlying the method in Section 4.1. Section 4.3 describes the solution of the index problem for snakes constructed by our method.

4.1 Outlines of the Construction

Our construction starts from a minimum-weight-d basis of some linear $[n,k,d]$-code $\mathcal{C}$, where n is the codeword length, k the dimension and d the minimum weight of $\mathcal{C}$. Many linear codes have minimum-weight basis. It has been proved in [31] that for any given triple $[n,k,d]$ for which it is known that there exists a linear code with these parameters, there is always a (possible non-equivalent) linear code with the same parameters which has a basis of vectors of weight d.

Let

$$\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k) \tag{4.1}$$

be a minimum-weight-d basis of $\mathcal{C}$. For every basis vector $\mathbf{b}_i \in \mathbf{B}$, we define its support

$$\sup \mathbf{b}_i = \{i_1, i_2, \ldots, i_d\} \tag{4.2}$$

where the integers $i_1, i_2, \ldots, i_d$ are all from the set $[0, n-1]$, and indicate the positions of the 1-bits of $\mathbf{b}_j$. The code $\mathcal{C}$ is a subspace of the vector space $V = GF(2)^n$ and we shall label the coordinates of the vectors of V, in particular those of $\mathcal{C}$, by $0, 1, 2, \ldots, n-1$, from left to right.

Now let

$$\overline{S}_k = 1, 2, 1, 3, 1, 2, 1, \ldots, 1, 2, 1, k, 1, 2, 1, 3, 1, 2, 1, \ldots, 1, 2, 1, k. \tag{4.3}$$

be the transition sequence of the standard Gray code $G(k)$ of word length k. In order to manipulate this sequence easier, we also write occasionally

$$\overline{S}_k = t_1, t_2, t_3, \ldots . t_{2^k}. \tag{4.4}$$

By applying the transition sequence $\overline{S}_k$ to the coordinate vectors of $\mathcal{C}$ with respect to basis $\mathbf{B}$ of (4.1) and starting with the zerovector, we obtain the following sequence of vectors

$$\mathbf{0}, \mathbf{b}_1, \mathbf{b}_1 + \mathbf{b}_2, \ldots, \mathbf{b}_{k-1}, \mathbf{b}_{k-1} + \mathbf{b}_k, \mathbf{b}_{k-1} + \mathbf{b}_k + \mathbf{b}_1, \mathbf{b}_{k-1} + \mathbf{b}_k + \mathbf{b}_1 + \mathbf{b}_2, \ldots, \mathbf{b}_k. \tag{4.5}$$

Because $G(k)$ is a *complete* Gray code, all 2^k vectors of $\mathcal{C}$ occur in (4.5) and because all $\mathbf{b}_i$ have weight d, each vector in (4.5) differs from its predecessor in precisely d bits, or stated equivalently, each codeword in the list is at Hamming distance d from the previous one. This property holds in cyclic sense, since $G(k)$ is cyclic. Here, the vector $\mathbf{b}_k$ is the predecessor of $\mathbf{0}$ (cf. also [39]). The list (4.5) constitutes the skeleton of the snake, or more generally of the circuit code, to be constructed.

To obtain the whole snake (circuit code), we have to change d bits any time when going from one word in the list (4.5) to the next one, in such a way that the separation property of a snake, i.e. no chords, is satisfied. To this end, we have to prescribe in which order the d bits in the RHS of (4.2) have to be changed in case that two successive codewords differ by $\mathbf{b}_i$, $1 \le i \le k$. One way to do this is to define for each basis vector $\mathbf{b}_i$ with support (4.2), an ordered set

$$B_i = (i_1, i_2, \ldots, i_d), \qquad 1 \le i \le k \tag{4.6}$$

indicating that first bit i_1 is to be changed, next bit i_2, etc.

In [10] linear $[n, k, 2]$-codes were considered with a minimum-weight-2 basis satisfying some additional properties which give rise to special distance preserving ($<m, n>$-) codes. In this thesis, we apply the method for $d = 4$. For this case, we shall describe our method in more detail.

Let $\mathcal{C}$ be some $[n, k, 4]$-code with basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k)$ as in (4.1), such that $\|\mathbf{b}_i\| = 4$, for $1 \le i \le k$. We assume that these basis vectors are ordered with respect to some, still unspecified, criterion giving rise to the ordered basis (4.1). Using the notation (4.4) for the transition sequence $\overline{S}_k$, we can define an ordered list for the codewords of $\mathcal{C}$ in the following way.

Starting from the zeroword $\mathbf{c}_0 = \mathbf{0}$, we define the codewords of $\mathcal{C}$ recursively (cf. (4.5))

$$\mathbf{c}_0 := \mathbf{0}, \mathbf{c}_{i+1} = \mathbf{c}_i + \mathbf{b}_{t_i}, \qquad 0 \le i < 2^k - 1. \tag{4.7}$$

Because of the properties of the Gray code and because of the constant weight 4 of all vectors $\mathbf{b}_i$, the lists (4.5) and (4.7) are complete lists of the 2^k words of $\mathcal{C}$, such that each codeword is at Hamming distance 4 from the previous one. Moreover, this last property holds cyclically. In order to arrive at a list of 4.2^k binary words satisfying the condition that each word differs from the previous one in precisely one bit, we transform $\mathbf{c}_i$ into $\mathbf{c}_{i+1}$ in (4.7) by changing the four bits of $\mathbf{b}_{t_i} = \mathbf{c}_i + \mathbf{c}_{i+1}$, one after another. This gives rise to intermediate words $\mathbf{w}_i^1$, $\mathbf{w}_i^2$ and $\mathbf{w}_i^3$, $0 \le i < 2^k - 1$. We call B_i in (4.6) an *ordered block*. The list of blocks

$$B_1, B_2, \ldots, B_k$$

corresponding to the sequence $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k$ will be denoted by $\mathcal{B}$. The order of the blocks is called the *external order* of the blocks. The sequence

$$\overline{S}_k(\mathcal{B}) = B_1 B_2 B_1 B_3 \ldots B_1 B_2 B_1 B_k B_1 B_2 B_1 B_3 \ldots B_1 B_2 B_1 B_k \tag{4.8}$$

where the blocks are arranged according to (4.3), can now be interpreted as a transition sequence of length $4 \cdot 2^n$, for binary words of length n, when the symbols B_i are replaced by the ordered sets (i_1, i_2, i_3, i_4), $1 \le i \le k$.

The order in the set of four integers is called the *internal order* of the block B_i. $1 \le i \le k$. In this thesis, this internal order of the blocks will be determined by a property which we call the *fixed-position property* and which will be defined in the next section. Applying (4.8),

starting from the zeroword **0**, provides us with the following list of words of length n

$$\mathbf{c}_0 := \mathbf{0},\ \mathbf{w}_0^1,\ \mathbf{w}_0^2, \mathbf{w}_0^3, \mathbf{c}_1,\ \mathbf{w}_1^1,\ \mathbf{w}_1^2, \mathbf{w}_1^3, \ldots,\ \mathbf{c}_{2^k-1},\ \mathbf{w}_{2^k-1}^1,\ \mathbf{w}_{2^k-1}^2,\ \mathbf{w}_{2^k-1}^3. \tag{4.9}$$

4.2 Necessary and Sufficient Conditions

In this section we shall prove that under certain conditions concerning the basis **B**, the outer order and the internal order of the blocks $B_i := \{i_1, i_2, i_3, i_4\}$, $1 \le i \le k$, the 2^{k+2} words of (4.9) constitute a snake. Not only the words in (4.9) must be different, but they also have to be at distance at least 2 (in cyclic sense) from each other when they are not neighbors.

Definition 4.1 (cf. [34, 35])

A block list $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *is said satisfying the fixed-position property if the internal order of the blocks of* $\mathcal{B}$ *is such that any integer of the set* $\{0, 1, \ldots, n-1\}$ *has a fixed position in each block of* $\mathcal{B}$ *in which it occurs. In this case, the integers* $0, 1, \ldots, n-1$ *can be partitioned into four sets* I_1, I_2, I_3 *and* I_4 *such that an integer* $i \in I_a$ *only occurs in position* a *in the blocks* $B_1, B_2, \ldots, B_k$, *for each* $a \in \{1, 2, 3, 4\}$.

Starting from Chapter 4 of this thesis, we label the positions in the binary words of length n from left to right by $0, 1, 2, \ldots, n-1$, (cf. also the remarks about labeling positions in binary words at the end of Section 1.2).

Lemma 4.1

Let $\mathcal{C}$ *be an* $[n, k, 4]$*-code with an ordered minimum-weight basis* $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k)$ *such that the corresponding list of blocks* $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *satisfies the fixed-position property. Let* $\mathbf{c} \in \mathcal{C}$ *be some codeword with* $W = \sup \mathbf{c}$. *Then the parity of* $|I_a \cap W|$ *is the same for all* $a \in \{1, 2, 3, 4\}$.

Proof. The proof follows immediately from the expression of **c** w.r.t. the basis **B**, i.e. $\mathbf{c} = \sum_{l=1}^{k} c_l \mathbf{b}_l$, $c_l \in \{0, 1\}$, and from the fixed-position property. □

In [11, 35] we discussed a number of examples of $[n, k, 4]$-codes which have bases satisfying this fixed-position property. In particular, the Reed-Muller codes $R(1, 3)$ and $R(2, 4)$ are such codes (cf. Section 5.4 and 5.5 of this thesis). Whether the sequence (4.8), which satisfies the fixed-position property, really defines a snake will actually depend on the specific choice of the basis vectors and of the external order of the blocks in $\mathcal{B}$. We shall derive some conditions which are necessary and/or sufficient for the external order of the blocks to generate a snake.

In the remaining part of this Chapter we shall consider subsequences of (4.8) of type

$$\mathcal{T} = B_i, \mathcal{T}', B_j, \qquad 1 \le i, j \le k \tag{4.10}$$

More in particular, we shall consider the contents $c(\mathcal{T})$ of such subsequences (cf. Def. 2.1). Since subsequence (4.10) consists of complete blocks, its content is the support of some codeword $\mathbf{c} \in \mathcal{C}$. We therefore define

$$W := \sup \mathbf{c} = c(\mathcal{T}).$$

If $i = 1, j > 1$ or $i > 1, j = 1$, $\mathbf{c}$ is the sum of an even number of basis vectors $\mathbf{b}_l \in \mathbf{B}$, whereas in all other cases, $\mathbf{c}$ is the sum of an odd number of basis vectors (cf. Theorem 2.5, (*ii*), (*iii*)). On the other hand, if $\mathbf{c}$ is some codeword of $\mathcal{C}$, it can be written as the sum of a number of basis vectors of basis $\mathbf{B}$, i.e.

$$\mathbf{c} = \sum_{l \in X} \mathbf{b}_l \,, \tag{4.11}$$

where the index set X is some subset of $\{1, 2, \ldots, k\}$. We define for our convenience, just as in Chapter 2,

$$i_0 := \min X. \tag{4.12}$$

Using Theorem 2.5, we can write for (4.11) in the case $1 < i < j$ that

$$\mathbf{c} = \mathbf{b}_{i_0} + \sum_{l \ge i_0 + 1} \mathbf{b}_l \,, \tag{4.13}$$

with $i_0 = i - 1$ and where the summation index l now runs through the index set

$$X \setminus \{i_0\}.$$

We shall write $\mathbf{e}_j$, $0 \le j < n$, for the unit vectors in $GF(2)^n$ with $(\mathbf{e}_j)_i = \delta_{ij}$, where i runs from 0 until $n - 1$ from left to right, e.g. $B_i = (i_1, i_2, i_3, i_4)$ corresponds to the basis vector

$$\mathbf{b}_i = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_{i_4}.$$

For our convenience, we shall sometimes assume, when studying sequence (4.10), that $i \le j$. In order to cover all possibilities, we then also have to take into account sequences $B_j, \mathcal{T}', B_i$. We now prove a necessary and sufficient criterion for a block list $\mathcal{B}$ to generate a snake.

Theorem 4.2

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a block list corresponding to an ordered minimum-weight basis of an $[n,k,4]$-code $\mathcal{C}$ satisfying the fixed-position property. Then the sequence $\overline{S}_k(\mathcal{B})$ of (4.8) *is the transition sequence of a snake of word length n and of range 2^{k+2} if and only if there is no codeword $\mathbf{c} \in \mathcal{C}$, as expressed by* (4.13)*, such that its support W is one of the following sets:*

(i) $W = \{i_1, i_2, p_3, j_4\}$;

(ii) $W = \{i_1, q_2, j_3, j_4\}$;

(iii) $W = \{j_1, j_2, p_3, i_4\}$;

(iv) $W = \{j_1, q_2, i_3, i_4\}$,

for $1 < i < j \le k$, *and for any p and q with* $1 \le p, q \le k$, *and where* p_2 *and* q_3 *are elements of blocks* B_p *and* B_q, *respectively.*

Proof. Let $\mathcal{S}$ be the list of words generated by $\overline{S}_k(\mathcal{B})$ as a transition sequence, starting from the zeroword **0**. Let **x** and **y** be two words of $\mathcal{S}$. We can write, w.l.o.g., $\mathbf{x} = \mathbf{c}' + \mathbf{z}'$ and $\mathbf{y} = \mathbf{c}'' + \mathbf{z}''$, where $\mathbf{c}', \mathbf{c}'' \in \mathcal{C}$ and where $\mathbf{z}'$ is one of the vectors $\mathbf{0}, \mathbf{e}_{i_1}, \mathbf{e}_{i_1} + \mathbf{e}_{i_2}, \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3}$, and $\mathbf{z}''$ is one of the vectors $\mathbf{0}, \mathbf{e}_{j_4}, \mathbf{e}_{j_3} + \mathbf{e}_{j_4}, \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, for some $i, j \in \{1, 2, \ldots, k\}$.

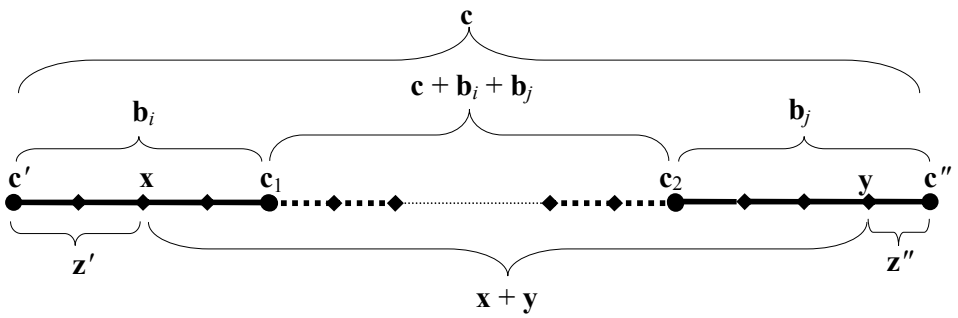

Fig. 4.1

We now partition the sequence $\overline{S}_k(\mathcal{B})$, for fixed values i and j, according to

$$\overline{S}_k(\mathcal{B}) = \mathcal{T}'', B_i, \mathcal{T}', B_j, \mathcal{T}'''$$

(this partition is not unique, in general), and assume that $\mathbf{c} = \mathbf{c}' + \mathbf{c}''$ is the codeword of $\mathcal{C}$ that corresponds to $\mathcal{T} := B_i, \mathcal{T}', B_j$, i.e. $W = \sup \mathbf{c} = c(\mathcal{T})$. If $\mathcal{T}'$ is empty then either $j > 1$, $i = 1$ or $i > 1$, $j = 1$ and it is obvious that $l(\mathbf{x}, \mathbf{y}) > 1$ is equivalent to $d(\mathbf{x}, \mathbf{y}) > 1$.

In the remaining part of this proof, we assume that $\mathcal{T}'$ is not empty. For similar reasons, we assume that the sublist $\mathcal{T}''', \mathcal{T}''$ is not empty (remember that $\mathcal{S}$ is a circular list).

We shall first prove that if the conditions of the Theorem hold, the list $\mathcal{S}$ is a snake.

A. Assume $d(\mathbf{x}, \mathbf{y}) = 0$.

It follows that $\mathbf{c} = \mathbf{c}' + \mathbf{c}'' = \mathbf{z}' + \mathbf{z}''$. The assumption $\mathcal{T}''', \mathcal{T}'' \neq \varnothing$ implies that $\mathbf{c} \neq \mathbf{0}$. Since $\mathbf{c} \in \mathcal{C}$, it follows that $\|\mathbf{c}\|$ equals 4 or 6. More in particular, due to the possibilities for $\mathbf{z}'$ and $\mathbf{z}''$, we have the following possible expressions for **c**.

(*a*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*b*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*c*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_{j_4}$;

(*d*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$.

In cases (*a*), we would have $\sup \mathbf{c} + \mathbf{b}_j = \{i_1, j_1\}$, and hence $\|\mathbf{c} + \mathbf{b}_j\| = 2$, violating the minimum distance 4 of $\mathcal{C}$, Similarly in cases (*e*) and (*d*), we would have $\|\mathbf{c} + \mathbf{b}_i\| = 2$ and $\|\mathbf{c} + \mathbf{b}_i + \mathbf{b}_j\| = 2$, respectively, which gives rise to the same contradiction. With respect to Case

(b), we remark that $\mathbf{c}$ is the sum of an odd number of basis vectors as a consequence of the fixed-position property, and therefore $i = j$ or $i > 1, j > 1$ or both (cf. Theorem 2.5). Now $i = j$ in (b) yields $\mathbf{c} = \mathbf{b}_i$ which contradicts Theorem 2.5 (*iii*). For $i \neq j$, a word of type (b) can not occur because of the conditions of the Theorem (take $p = j$ in (i) or $q = i$ in (*ii*)).

B. Assume $d(\mathbf{x}, \mathbf{y}) = 1$.

Now, $\mathbf{x} + \mathbf{y} = \mathbf{e}_t$ for some $t \in \{1, 2, \ldots, n\}$ and it follows that $\mathbf{c} = \mathbf{z}' + \mathbf{z}'' + \mathbf{e}_t$. Since $\mathbf{c} \in \mathcal{C}$, we have again that $\|\mathbf{c}\|$ is equal to 4 or 6. The only possibilities for $\mathbf{c}$ with $\|\mathbf{c}\| = 4$ are (cf. (4.10))

(e) $\mathbf{c} = \mathbf{e}_t + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $\quad t \in \mathrm{I}_1 \setminus \{i_1, j_1\}$;

(f) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_t + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $\quad t \in \mathrm{I}_2 \setminus \{i_2, j_2\}$;

(g) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_t + \mathbf{e}_{j_4}$, $\quad t \in \mathrm{I}_3 \setminus \{i_3, j_3\}$;

(h) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_t$, $\quad t \in \mathrm{I}_4 \setminus \{i_4, j_4\}$.

Cases (e) and (h) do not occur, since then we would have $\|\mathbf{c} + \mathbf{b}_j\| = 2$ and $\|\mathbf{c} + \mathbf{b}_i\| = 2$, respectively, which violates the minimum distance 4 of $\mathcal{C}$. Cases (f) and (g) do not occur because of the conditions of the Theorem.

The possibilities for $\mathbf{c}$ with $\|\mathbf{c}\| = 6$ are

(i) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_t + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $\quad t \in \mathrm{I}_3 \setminus \{i_3\}$;

(j) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_t + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $\quad t \in \mathrm{I}_2 \setminus \{j_2\}$.

It will be clear that any choice for t will contradict Lemma 4.1. So, we have proved now the *if* part of the Theorem.

Now we shall prove the *only-if* part. Let $\mathcal{S}$ be a snake. Assume that the conditions of the Theorem do not hold. Then there exists a $\mathbf{c} \in \mathcal{C}$ (cf. eq. (4.13))

$$\mathbf{c} = \sum_{l \in X} \mathbf{b}_l = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l ,$$

for some $X \subseteq \{1, 2, \ldots, k\}$, $\min X = i - 1$, such that its contents W is equal to one of the sets (i) – (iv) mentioned in the Theorem.

E.g., let $W = \{i_1, i_2, p_3, j_4\}$ with $1 < i < j$. From Theorem 2.7(i), we know that the transition sequence $\overline{S}_k$ of the standard Gray code $G(k)$ contains a subsequence $T = i, T', j$ with $c(T) = X$ (remember that the codewords of $\mathcal{C}$ are ordered with respect to the standard Gray code $G(k)$ of length k, which is the dimension of $\mathcal{C}$, whereas n stands for the length of codewords in $\mathcal{C}$, contrary to the role of n in Chapter 2).

It follows that $\overline{S}_k$ ($\mathcal{B}$) contains a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ with

$$c(\mathcal{T}) = W = \{i_1, i_2, p_3, j_4\}, \qquad 1 < i < j.$$

If $p_3 = i_3$, we take $\mathbf{z}' = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3}$ and $\mathbf{z}'' = \mathbf{e}_{j_4}$ (cf. the beginning of this proof), and we obtain words $\mathbf{x} = \mathbf{c}' + \mathbf{z}'$, $\mathbf{y} = \mathbf{c}'' + \mathbf{z}''$ with mutual distance

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} + \mathbf{y}\| = \|\mathbf{c} + \mathbf{z}' + \mathbf{z}''\| = 0.$$

This contradicts our assumption that $\mathcal{S}$ is a snake. Similarly, $p_3 = j_3$ gives $d(\mathbf{x}, \mathbf{y}) = 0$. If $p_3 \neq i_3$, j_3, we take $\mathbf{z}' = \mathbf{e}_{i_1} + \mathbf{e}_{i_2}$ and $\mathbf{z}'' = \mathbf{e}_{j_4}$ giving rise to

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{e}_{p_3}\| = 1,$$

which also contradicts the assumption that $\mathcal{S}$ is a snake.

A similar argument can be given in case (*ii*). In cases (*iii*) and (*iv*) we consider the codeword $\mathbf{c}' = \mathbf{c} + \mathbf{b}_i + \mathbf{b}_j$ which can also be expressed in the form of equation (4.13) since $i > i - 1$ and $j > i - 1$. Similarly as in cases (*i*) and (*ii*), we now can derive contradictions. Therefore, the conditions of the Theorems are necessary as well. □

The four cases in Theorem 4.2 can be replaced just by (*i*) and (*ii*) under the wider condition that $1 < i, j \leq k$.

We shall show that an immediate consequence of Theorem 4.2 is, that if the blocks in $\mathcal{B}$ are *well-ordered*, the sequence $\overline{S}_k(\mathcal{B})$ of (1) generates a snake. The notion of a well-ordered basis which satisfies the fixed-position property reads as follows.

Let

$$\mathcal{B} = (B_1, B_2, \ldots, B_k)$$

be a list of blocks such that the internal order of the blocks satisfies the fixed-position property (cf. also [34])

Definition 4.2

$\mathcal{B}$ is called well-ordered if

(i) there is only one integer in I_1 – say m_1 – which occurs in more than one block, and the blocks are ordered such that $1_1, 2_1, \ldots, m_1$ are all distinct and

$$m_1 = (m + 1)_1 = \ldots = k_1,$$

where i_1 is the first integer of block B_i, $1 \leq i \leq k$;

(ii) for all i with $m \leq i < k$ at least one of the integers i_2 and i_4 does not occur in any of the blocks B_l, $l > i$.

More generally, we shall call a block list well-ordered, if it can be transformed into a list satisfying (i) and (ii) above by a permutation of the integers $0, 1, 2, \ldots, n - 1$.

Remark.

In practice, we shall apply the last rule in Definition 4.2 such that the order of integers in I_1 is

$$1_1 > 2_1 > \ldots > m_1 = (m + 1)_1 = \ldots = k_1 = 0. \tag{4.14}$$

In this case, we say that the blocks are *ordered lexicographically with respect to the integers in* I_1. Occasionally, we shall say that the blocks of some block list are *ordered lexicographically* if the condition (*i*) of Definition 4.2 is satisfied. It will turn out in Chapter 5 that a block containing the integer 0 can be interpreted as a linear subspace of $GF(2)^n$.

Corollary 4.3

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *be a block list of an* $[n, k, 4]$*-code which satisfies the fixed-position property. If* $\mathcal{B}$ *is well-ordered, then* $\overline{S}_k(\mathcal{B})$ *is the transition sequence of a snake.*

Proof. Suppose that $\mathcal{S}$ is not a snake. Then we have from Theorem 4.2 that there exists a codeword

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

(cf. (4.13)) such that $W = \sup \mathbf{c}$ is equal to one of the sets (*i*) – (*iv*). Suppose for example that $W = \{i_1, i_2, p_3, j_4\}$, $1 < i < j$ (Case (*i*)).

For $1 < i \leq m$, it follows from the well-ordering of the blocks of $\mathcal{B}$ that $(i-1)_1$ does not occur anymore in any of the blocks B_l, $l \geq i$. Hence, $(i-1)_1 \neq i_1 \in W$. On the other hand, because of (4.13), we have that $(i-1)_1 = i_1 \in W$ which contradicts $(i-1)_1 \neq i_1$.

For $i > m$, $(i-1)_1 = i_1 = \ldots = k_1$ and (cf. Definition 4.2) at least one of the integers $(i-1)_2$ or $(i-1)_4$ does not occur anymore in a block B_l, $l \geq i$. If $(i-1)_2$ does not occur anymore, then $(i-1)_2 \neq l_2$, $l \geq i$. In particular, $(i-1)_2 \neq i_2$. On the other hand from (4.13), we have $(i-1)_2 \in W$. But this implies, by the fixed-position property and the fact that $W = \{i_1, i_2, p_3, j_4\}$, that $(i-1)_2 = i_2$, a contradiction.

Similarly, if $(i-1)_4$ does not occur anymore, then $(i-1)_4 \neq j_4$, because $j > i$. On the other hand, from (4.13) we have $(i-1)_4 \in W$. But this implies, by the fixed-position property and the fact that $W = \{i_1, i_2, p_3, j_4\}$, $(i-1)_4 = j_4$, a contradiction with the previous inequality. We conclude that $W \neq \{i_1, i_2, p_3, j_4\}$.

Now suppose $W = \{i_1, q_2, j_3, j_4\}$. For $1 < i \leq m$, we can eliminate this possibility in precisely the same way as Case (*i*). Suppose $i > m$. By the same argument as before, we can show that if $(i-1)_4$ does not occur, we must have $(i-1)_4 = j_4$, which is a contradiction since $j > i$. So, $(i-1)_2$ does not occur anymore. It follows that $q_2 = (i-1)_2$. Furthermore, since $j > i$, we also have $i_1 = j_1 = m_1$ and therefore $W = \{j_1, q_2, j_3, j_4\}$, with $q_2 = (i-1)_2 \neq j_2$. However, this implies $\|\mathbf{c} + \mathbf{b}_j\| = 2$, which contradicts the minimum weight of $\mathcal{C}$ being 4. So, Case (*ii*) is not possible.

The proofs for Cases (*iii*) and (*iv*) are similar to the proofs for Cases (*i*) and (*ii*), respectively. □

In case a block list $\mathcal{B}$ is only ordered lexicographically according to condition (*i*) of Definition 4.2 and the supplementing remark, and not necessarily satisfies condition (*ii*), we can formulate Theorem 4.2 in a simpler way.

Theorem 4.4

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *be a block list corresponding to an ordered minimum-weight basis of an* $[n,k,4]$*-code* $\mathcal{C}$ *satisfying the fixed-position property. Let furthermore* $\mathcal{B}$ *be ordered lexicographically with respect to the integers in* I_1 *according to Definition* 4.2*, then the sequence* $\overline{S}_k(\mathcal{B})$ *of* (4.8) *is the transition sequence of a snake if and only if there is no codeword* $\mathbf{c} \in \mathcal{C}$*, as expressed by* (4.13)*, such that its support W is one of the following sets*

(i) $W = \{m_1, i_2, p_3, j_4\}$*;*

(ii) $W = \{m_1, j_2, q_3, i_4\}$,

for $m < i < j \le k$*, and for any p and q with* $m - 1 \le p, q \le k$.

Proof. The necessity of the condition follows immediately from Theorem 4.2. Suppose that the list of codewords generated by $\overline{S}_k(\mathcal{B})$ is not a snake. Then we have from Theorem 4.2 that there exists a codeword (cf. eqs. (4.12) and (4.13))

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \ge i} \mathbf{b}_l \,,$$

for some i, $1 < i \le k$, such that $W = \sup \mathbf{c}$ is equal to one of the sets (*i*) – (*iv*) of Theorem 4.2.

For $1 \le i \le m$, it follows from the lexicographic order of the blocks in $\mathcal{B}$ that $(i-1)_1$ does not occur anymore in blocks B_l, $l \ge i$. Hence, because of (4.13), we have that $(i-1)_1 \in W$ which contradicts $(i-1)_1 \ne m_1$.

Now, let $i > m$. If W is of type (*ii*) in Theorem 4.2, we have (because $j > i > m$)

$$W = \{m_1, q_2, j_3, j_4\} = \{j_1, q_2, j_3, j_4\}.$$

Since the minimum distance in $\mathcal{C}$ is 4, it follows that $q_2 = j_2$ and so $\mathbf{c} = \mathbf{b}_j$. However, this contradicts expression (4.13), because $j > i - 1$. Similarly, if W is of type (*iv*) in Theorem 4.2, we obtain that $\mathbf{c} = \mathbf{b}_i$, which also contradicts expression (4.13). So, we are left with types (*i*) and (*iii*) of Theorem 4.2, which proves the sufficiency of this Theorem, because $i_1 = j_1 = m_1$. □

Example 4.1.

We first give the simple example of a snake of range 64 in Q_8, generated by four vectors that are represented by the list of four blocks $\mathcal{B} = (B_1, B_2, B_3, B_4)$ with

$$B_1 = (0,2,4,5), \quad B_2 = (1,2,4,6), \quad B_3 = (1,3,7,6), \quad B_4 = (1,3,4,5).$$

These blocks correspond to four independent vectors of weight 4 in $GF(2)^8$, which are the basis vectors of an [8,4,4]-code. One can easily verify that $\mathcal{B}$ satisfies the fixed-position property. Moreover, the list is ordered lexicographically. According to Corollary 4.3 or to

Theorem 4.4, the sequence $\overline{S}_4(\mathcal{B})$ generated by the list $\mathcal{B}$, is the transition sequence of a snake of length 2^6 in Q_8.

The following list provides all words of the snake, together with the integers of its transition sequence that determine the next word. We also add the indices of the words starting from 0 until 63.

0. 00000000; (0);	1. 10000000; (2);	2. 10100000; (4);	3. 10101000; (5);
4. 10101100; (1);	5. 11101100; (2);	6. 11001100; (4);	7. 11000100; (6);
8. 11000110; (0);	9. 01000110; (2);	10. 01100110; (4);	11. 01101110; (5);
12. 01101010; (1);	13. 00101010; (3);	14. 00111010; (7);	15. 00111011; (6);
16. 00111001; (0);	17. 10111001; (2);	18. 10011001; (4);	19. 10010001; (5);
20. 10010101; (1);	21. 11010101; (2);	22. 11110101; (4);	23. 11111101; (6);
24. 11111111; (0);	25. 01111111; (2);	26. 01011111; (4);	27. 01010111; (5);
28. 01010011; (1);	29. 00010011; (3);	30. 00000011; (4);	31. 00001011; (5);
32. 00001111; (0);	33. 10001111; (2);	34. 10101111; (4);	35. 10100111; (5);
36. 10100011; (1);	37. 11100011; (2);	38. 11000011; (4);	39. 11001011; (6);
40. 11001001; (0);	41. 01001001; (2);	42. 01101001; (4);	43. 01100001; (5);
44. 01100101; (1);	45. 00100101; (3);	46. 00110101; (7);	47. 00110100; (6);
48. 00110110; (0);	49. 10110110; (2);	50. 10010110; (4);	51. 10011110; (5);
52. 10011010; (1);	53. 11011010; (2);	54. 11111010; (4);	55. 11110010; (6);
56. 11110000; (0);	57. 01110000; (2);	58. 01010000; (4);	59. 01011000; (5);
60. 01011100; (1);	61. 00011100; (3);	62. 00001100; (4);	63. 00000100. (5).

Fig. 4.2

Notice that the sixteen words in the first column constitute the linear [8,4,4]-code being the 'back-bone' of the snake. These are the snake words that have an index divisible by 4. We also observe that only two of the eight different integers of the transition sequence occur in that column. This fact is due to the fixed position property. A similar observation can be made with respect to the other three columns.

As long as we do not change the order of the blocks that generate a snake in Q_n, we can omit any block from the list of blocks in order to get a smaller snake in $Q_{n'}$ with $n' \leq n$. This is because the omission of one block does not harm the well-ordering of the list.

E.g. by omitting B_1, one obtains a list which generates a snake $\mathcal{S}$ of length 2^5 in Q_7 (cf. [34, Section 3]), since the integer 0 does not occur in any of the remaining blocks. More precisely, we can speak of the snake $\mathcal{S}$ of range 2^5 in a subgraph of Q_8, which is equivalent to Q_7 as we can observe from the following list of words of $\mathcal{S}$.

0. 00000000; (1);	1. 01000000; (2);	2. 01100000; (4);	3. 01101000; (6);
4. 01101010; (1);	5. 00101010; (3);	6. 00111010; (7);	7. 00111011; (6);
8. 00111001; (1);	9. 01111001; (2);	10. 01011001; (4);	11. 01010001; (6);
12. 01010011; (1);	13. 00010011; (3);	14. 00000011; (4);	15. 00001011; (5);
16. 00001111; (1);	17. 01001111; (2);	18. 01101111; (4);	19. 01100111; (6);
20. 01100101; (1);	21. 00100101; (3);	22. 00110101; (7);	23. 00110100; (6);
24. 00110110; (1);	25. 01110110; (2);	26. 01010110; (4);	27. 01011110; (6);
28. 01011100; (1);	29. 00011100; (3);	30. 00001100; (4);	31. 00000100; (5);

Fig. 4.3

Clearly, we can ignore or puncture the first position to get a snake of the same range in Q_7. Since 7 occurs only once in B_3, omitting B_3 and puncturing the last position will also produce another snake in Q_7, as shown in the following list.

0. 0000000 (0)	1. 0000001 (2)	2. 0000101 (4)	3. 0010101 (5)
4. 0110101 (1)	5. 0110111 (2)	6. 0110011 (4)	7. 0100011 (6)
8. 1100011 (0)	9. 1100010 (2)	10. 1100110 (4)	11. 1110110 (5)
12. 1010110 (1)	13. 1010100 (3)	14. 1011100 (4)	15. 1001100 (5)
16. 1101100 (0)	17. 1101101 (2)	18. 1101001 (4)	19. 1111001 (5)
20. 1011001 (1)	21. 1011011 (2)	22. 1011111 (4)	23. 1001111 (6)
24. 0001111 (0)	25. 0001110 (2)	26. 0001010 (4)	27. 0011010 (5)
28. 0111010 (1)	29. 0111000 (3)	30. 0110000 (4)	31. 0100000. (5)

Fig. 4.4

We observe that omitting block B_4 from $\mathcal{B}$ will not give a snake in Q_7 because all integers of the last block also occur in other blocks.

Now consider the following block list $\mathcal{B}'$:

$$B_1 = (1, 2, 3, 4),$$
$$B_2 = (5, 6, 3, 4),$$
$$B_3 = (1, 6, 7, 8).$$

The list satisfies the fixed-position condition and corresponds to a basis of an [8, 3, 4]-code $\mathcal{C}$. The sequence $\overline{S}_3(\mathcal{B}')$ is not the transition sequence of a snake $\mathcal{S}'$, since it contains a subsequence $\mathcal{T} = B_2, B_1, B_3$ which corresponds to the codeword $\mathbf{c} = \mathbf{b}_2 + \mathbf{b}_1 + \mathbf{b}_3 \in \mathcal{C}$ with support $W = \{5, 2, 7, 8\}$. Omitting from $\mathcal{T}$ the first integer 2_1 (= 5) and the last two integers 3_3 (= 7) and 3_4(= 8) shows that there are two words $\mathbf{x}$ and $\mathbf{y}$ in the list $\mathcal{S}$ which are no neighbors of each other, and which have Hamming distance 1. More in particular, we can write $\mathbf{x} = \mathbf{c}' + \mathbf{e}_{2_1}$, $\mathbf{y} = \mathbf{c}'' + \mathbf{e}_{3_3} + \mathbf{e}_{3_4}$, $\mathbf{c} = \mathbf{c}' + \mathbf{c}''$. Indeed, the necessary condition (*ii*) of Theorem 4.2 is not satisfied for $i = 2, j = 3$.

This example demonstrates that if we generalize the well-ordering condition in the sense that for all i, $1 \le i < k$, at least one of the integers i_1, i_2 or i_4 does not occur in any of the blocks B_l, $l \ge i$, the condition is no longer sufficient for $\mathcal{S}$ being a snake.

4.3 The Index Problem of a Snake in Q_n

We number the words of a snake $\mathcal{S}$ in Q_n of length 2^l by an index i which ranges from 0 until $2^l - 1$. Now, given some value i for this index, the problem arises to determine the corresponding word $\mathbf{w}_i$ in the snake. We call this problem, together with the inverse problem of determining the index of a given word $\mathbf{w}_i$ in $\mathcal{S}$, *the index problem of* $\mathcal{S}$. We assume that $\mathcal{S}$ is constructed by the method discussed in Section 4.2. So, $l = k + 2$ where k is the dimension of the underlying linear code $\mathcal{C}$.

Before studying the index problem for our snakes, we first remind the reader of the previous conventions adopted by us in this thesis (and in [34, 35]) with respect to the labeling of the words in cyclic lists and of the bits within a word.

The index x of the word $\mathbf{g}_x$ of the cyclic standard Gray code $G(k)$ of word length k runs from 0 until $2^k - 1$. The coordinate or bits g_{xj} of $\mathbf{g}_x$ (shortly denoted by g_j if the index x is irrelevant) are labeled from 1 until k, *from right to left*, $k \ge i \ge 1$, just like in Chapter 2 where we studied $G(n)$ and where i runs from 1 to k. Inherent to the index notation for $G(k)$ is that its transition sequence contains integers from $\{1, 2, \ldots, k\}$. It follows that the basis vectors of the linear $[n, k, 4]$-code, which are the building stones of the snake(s) to be constructed, are labeled as $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k$, and so are the corresponding blocks $B_1, B_2, \ldots, B_k$.

The integers in a block $B_i = (i_1, i_2, i_3, i_4)$, $1 \le i \le k$, are taken from the set $\{0, 1, \ldots, 2^m - 1\}$ with $2^m = n$. This choice is due to the definitions given in [35] concerning the Euclidean geometry EG(m, 2). Here, the bits of the words of a snake in Q_n, based on our construction, are labeled from 0 until $n - 1$ ($= 2^m - 1$). The order of labeling may be chosen at will, from right to left as was introduced in [35, Section 3.1], or from left to right as will be applied in this thesis. Finally with $L = 2^{k+2}$ (cf. (2.18)), the index of the snake words runs from 0 (corresponding to the zero word) to $2^l - 1$ with $l = k + 2$, as was already used in expression (4.9).

Recall that given an index x represented by a binary k-tuple $\mathbf{x} = x_k, x_{k-1}, \ldots, x_1$, the rule to compute $\mathbf{g}_x$ as stated in Theorem 2.1,

$$g_i = x_i + x_{i+1} \pmod 2, \tag{4.15}$$

can symbolically be written as $\mathbf{g}_x = \mathbf{x} \oplus \lfloor \mathbf{x}/2 \rfloor$, with $x_{k+1} = 0$, where $\oplus$ stands for bitwise addition modulo 2 (exclusive-or operator).

An alternative expression for $\mathbf{g}_x$ is

$$\mathbf{g}_x = \sum_{j=1}^{k} (x_j + x_{j+1})\mathbf{f}_j = \sum_{j \in J_x} \mathbf{f}_j \,, \tag{4.16}$$

where $J_x := \sup \mathbf{g}_x$ and the $\mathbf{f}_j$, $1 \le j \le k$, are the unit vectors (words) in $GF(2)^k$, such that the i-th bit of $\mathbf{f}_j$ is $(\mathbf{f}_j)_i = \delta_{ij}$, $1 \le i \le k$. Because of our conventions w.r.t. the labeling of the bit positions in $\mathbf{g}_x$, the only 1-bit of $\mathbf{f}_j$ is at the j-th position *from the right*.

We now return to the index problem of a snake $\mathcal{S}$ of length 2^{k+2} constructed by our method, based on an ordered weight-4 basis $\mathcal{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k)$ of a linear $[n, k, 4]$-code $\mathcal{C}$. Let i be some integer with $0 \le i < 2^{k+2}$. If i is a multiple of 4, say $i = 4x$, then $\mathbf{w}_i$ is equal to the codeword $\mathbf{c}_x$ of $\mathcal{C}$, where x is the index of this word when listed according to the standard Gray code $G(k)$ which has as consequence that any two successive words of $\mathcal{C}$ in the list have Hamming distance 4 (cf. eq. (4.9)). It follows that $\mathbf{w}_i = \mathbf{c}_x$ can be determined from x by first computing $\mathbf{g}_x \in G(k)$ and next writing

$$\mathbf{w}_i = \sum_{j \in J_x} \mathbf{b}_j ,$$

where J_x is defined by eq. (4.16).

In the general case, we write for the index i

$$i = 4x + z, \qquad z \in \{0, 1, 2, 3\}. \tag{4.17}$$

The snakeword $\mathbf{w}_i$ can now be written as $\mathbf{w}_i = \mathbf{c}_x + \mathbf{z}$, where $\mathbf{z}$ is equal to one of $\mathbf{0}$, $\mathbf{e}_{t_1}$, $\mathbf{e}_{t_1} + \mathbf{e}_{t_2}$ and $\mathbf{e}_{t_1} + \mathbf{e}_{t_2} + \mathbf{e}_{t_3}$. The integers t_j, $j = 1, 2, 3$, are from block $B_t = \{t_1, t_2, t_3, t_4\}$, which corresponds to $\mathbf{b}_t = \mathbf{c}_{x+1} - \mathbf{c}_x$.

Hence, we formulate the following algorithm to compute the i-th word in the snake $\mathcal{S}$ defined by the transition sequence

$$\overline{S}_k(\mathcal{B}) = B_1\, B_2\, B_1 \ldots B_1\, B_k\, B_1\, B_2 \ldots B_k.$$

Algorithm

1. Write $i = 4x + z$. $\qquad z \in \{0, 1, 2, 3\}$.
2. Compute $\mathbf{g}_x = \mathbf{x} + \lfloor \mathbf{x}/2 \rfloor$ and let $J_x = \sup \mathbf{g}_x$.
3. Compute $\mathbf{g}_{x+1} = \mathbf{y} + \lfloor \mathbf{y}/2 \rfloor$, where $y = x + 1$ and let $J_y = \sup \mathbf{g}_y$.
4. Determine $t = J_x \oplus J_{x+1}$ (t is the symmetric difference of J_x and J_{x+1}).
5. $\mathbf{w}_i = \sum_{j \in J_x} \mathbf{b}_j + \sum_{j=1}^{z} \mathbf{e}_{t_j}$ with $\sum_{j=1}^{0} \mathbf{e}_{t_j} := \mathbf{0}$.

In this algorithm, the binary vector $\mathbf{x}$ is identical with the binary representation of the integer x, while $\lfloor \mathbf{x}/2 \rfloor$ stands for the binary representation of the integer $\lfloor x/2 \rfloor$. The sign '+' represents the usual addition between two vectors of $GF(2)^k$ (We can drop the notation '$\oplus$' since in $GF(2)^k$, the meaning of '+' and of '$\oplus$' are the same).

Example 4.2.

The following list $\mathcal{B} = (B_1, B_2, B_3, B_4)$ of blocks corresponds to a minimum-weight basis of a [8, 4, 4]-code

$$B_1 = (0, 2, 4, 5),$$
$$B_2 = (1, 2, 4, 6),$$
$$B_3 = (1, 3, 7, 6),$$
$$B_4 = (1, 3, 4, 5).$$

We shall compute the snake words $\mathbf{w}_{52}$ and $\mathbf{w}_{54}$, by applying our algorithm. We write $i = 52 = 4 \cdot 13$. In the binary number system we have that 13 is represented by 1101. Hence,

$$\mathbf{g}_{13} = 1101 + 0110 = 1011$$

and so

$$\mathbf{w}_{52} = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_4 = 10011010.$$

Similarly, we have $\mathbf{g}_{14} = 1001$ and $\mathbf{w}_{56} = \mathbf{b}_1 + \mathbf{b}_4 = 11110000$. Therefore

$$t = 2$$

and since $B_2 = \{1, 2, 4, 6\}$ corresponding to $\mathbf{b}_2 = \mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_4 + \mathbf{e}_6$, it follows that

$$\mathbf{w}_{54} = \mathbf{w}_{52} + \mathbf{e}_1 + \mathbf{e}_2 = 10011010 + \mathbf{e}_1 + \mathbf{e}_2 = 11111010$$

or

$$\mathbf{w}_{54} = \mathbf{w}_{56} + \mathbf{e}_4 + \mathbf{e}_6 = 11110000 + \mathbf{e}_4 + \mathbf{e}_6 = 11111010.$$

These results agree with the list S_8 (cf. [19] , Appendix C).

There is an alternative rule for the computation of the index value t which indicates the bit position which changes when going from $\mathbf{g}_x$ to $\mathbf{g}_{x+1}$ in $G(k)$. Actually, t is the bit position where the carry stops when adding 1 to the binary representation of $\mathbf{x}$. The rule can be inferred from the rule embodied by Steps 3 and 4 of the Algorithm. So, we can reformulate our algorithm in a more compact form.

Algorithm

1. Write $i = 4x + z$, $\quad z \in \{0, 1, 2, 3\}$.

2. Compute $\mathbf{g}_x = \mathbf{x} + \lfloor \mathbf{x}/2 \rfloor$ and let $J_x = \sup \mathbf{g}_x$.

3. Determine t by adding 1 to x and see where the carry stops.

4. $\mathbf{w}_i = \sum_{j \in J_x} \mathbf{b}_j + \sum_{j=1}^{z} \mathbf{e}_{t_j}$ with $\sum_{j=1}^{0} \mathbf{e}_{t_j} := \mathbf{0}$.

Example 4.3.

In Example 4.2, we have that $x = 13$. If we add 1 to 13 in the binary system, the carry stops at position 2.

5. Snakes in the Hypercubes Q_n. for $3 < n \leq 16$.

In this chapter, we shall discuss the construction of symmetric snakes in all hypercubes Q_n, $3 \leq n \leq 16$. As was pointed out in Chapter 4, an essential element of our method is a basis of a linear $[n, k, 4]$-code that satisfies the fixed-position property. Therefore, we show in Sections 5.1, 5.2 and 5.3 that the Reed-Muller code $R(r, m)$ with $r = m - 2$ is an appropriate code which satisfies all the required properties. Sections 5.4 and 5.5 contain examples of our construction for $m = 3$ and $m = 4$. Section 5.6 presents a sufficient condition for an ordered basis of an arbitrary $[n, k, 4]$-code to generate a snake.

5.1 Euclidean Geometries

As for the definitions and notation with respect to Euclidean geometries and their relation with Reed-Muller codes, we shall keep as close as possible to ([23, Ch.13]). Let $V = GF(q)^{m+1}$ be the vector space of dimension $m + 1$ over the field $GF(q)$, where q is some prime power. Let $\mathbf{x}$, $\mathbf{y}$ be two vectors of $V^* = V \setminus \{0\}$. The statement $\mathbf{x} = \lambda\mathbf{y}$, for some $\lambda \in GF(q)$, $\lambda \neq 0$, defines an equivalence relation on V^*, the classes of which are, by definition, the points of the *projective geometry* PG(m, q) (cf. [3]). We shall denote the equivalence classes by $\langle\mathbf{x}\rangle$, where $\mathbf{x} \in V^*$ is some element in this class. Hence, $\langle\mathbf{x}\rangle$ stands for all non-zero multiples of $\mathbf{x}$. A subspace $\langle U\rangle$ of PG(m, q) is the image of a subspace U of V under the map $\mathbf{x} \rightarrow \langle\mathbf{x}\rangle$.

For geometrical reasons one says that if U has dimension r, then $\langle U\rangle$ has dimension $r - 1$ (cf. [23, 35]). In particular, the dimension of PG(m, q) is m, and we write dim PG(m, q) = m. A hyperplane H of a projective geometry PG(m, q) is a subspace with dim $H = m - 1$.

The *Euclidean* or *affine geometry* EG(m, q) (or AG(m, q)), with $m \geq 1$ and q is a prime power, is obtained from the projective geometry PG(m, q) by deleting the points of an arbitrary hyperplane H. If one chooses H to be the plane consisting of all points $\langle(0, a_1, a_2, \ldots, a_m)\rangle \in$ PG(m, q), the remaining points of PG(m, q) can be labeled by $\langle(1, a_1, a_2, \ldots, a_m)\rangle$. By deleting the common 1, the q^m points of EG(m, q) can be labeled by the m-tuples $(a_1, a_2, \ldots, a_m)$, $a_i \in GF(q)$, and hence, these points can be considered as the vectors of the linear space $GF(q)^m$.

For practical reasons (cf. [35], Section 3.2]) we shall relabel the components of these vectors and write $(a_0, a_1, \ldots, a_{m-1})$. Addition and scalar multiplication in EG(m,q) is induced by the corresponding operations in $GF(q)^m$, $m \geq 1$. For this reason, m is said to be the dimension of EG(m, q).

From the definition of EG(m, q), we have that its points are the points of PG(m, q)

which are *not* in H. We also define the *lines* of EG(m, q) as those *lines* of PG(m, q) which are not contained in H. In general, the *r-dimensional subspaces* of EG(m, q) are those *r-dimensional subspaces* of PG(m, q) which are not in H. Incidence relations in EG(m, q) are induced by the incidence relations in PG(m, q) (cf. [23], App. B).

A subspace S of EG(m, q) is called a *flat*. One can easily show that a flat S is a *linear* subspace if and only if $\mathbf{0} = (0, 0, 0, \ldots, 0)$ is contained in S. Furthermore, a flat of dimension r is either a linear subspace of dimension r or a coset of such a linear subspace. Since it can easily be proved that a linear subspace of EG(m, q) of dimension r is algebraically isomorphic to EG(r, q), an r-dimensional flat is sometimes referred to as an EG(r, q), or also as an r-flat. We emphasize that the subset $\{\mathbf{0} := (0, 0, \ldots, 0)\}$ is a 0-dimensional subspace of EG(m, q), just like all other subsets containing only one point.

In the next section, we shall occasionally need the number of times that a certain Euclidean geometry is contained in a larger Euclidean geometry, or vice versa. These numbers are given by the following theorems, the proof of which can be found in [23].

Theorem 5.1 ([23], App. B, Theorem 5,6])

(i) The number of EG(r, q) *in* EG(m, q) *is equal to*

$$q^{m-r}\begin{bmatrix} m \\ r \end{bmatrix}_q .$$

(ii) If in EG(m, q) *one has the inclusion*

$$R = \mathrm{EG}(r, q) \subseteq S = \mathrm{EG}(s, q),$$

where $r \geq 1$*, then the number of t-dimensional flats* T *with* $R \subseteq T \subseteq S$ *is equal to*

$$\begin{bmatrix} s-r \\ t-r \end{bmatrix}_q .$$

The symbol $\begin{bmatrix} m \\ r \end{bmatrix}_q$ stands for the Gaussian binomial coefficient defined by

$$\begin{bmatrix} m \\ r \end{bmatrix}_q = \frac{(q^m-1)(q^m-q)\ldots(q^m-q^{r-1})}{(q^r-1)(q^r-q)\ldots(q^r-q^{r-1})} .$$

Let S be some linear r-dimensional subspace of EG(m, q). Then S and its cosets are all pairwise disjoint. Consider a complete family of cosets of S, i.e.

$$\mathcal{P} := \{S + \mathbf{p}_0, S + \mathbf{p}_1, S + \mathbf{p}_2, \ldots, S + \mathbf{p}_{l-1}\} \tag{5.1}$$

with $\mathbf{p}_0 = \mathbf{0}, \mathbf{p}_1, \ldots, \mathbf{p}_{l-1} \in V$, such that

$$\bigcup_i (S + \mathbf{p}_i) = V.$$

Disjoint cosets (i.e. r-flats in EG(m, q)) will be called *parallel* subspaces or *parallel* flats in EG(m, q). Since the union of cosets in (5.1) contains all points of EG(m, q), we call $\mathcal{P}$ a

parallel system of flats. We also say that the parallel system $\mathcal{P}$ *covers* EG(m, q). If we consider the linear spaces V and S as additive groups, we can say that $\mathcal{P}$ is the quotient group

$$\mathcal{P} = V/S. \tag{5.2}$$

Obviously, the number l of parallel r-flats in EG(m, 2) is equal to

$$|\mathcal{P}| = 2^{m-r}.$$

For more properties of EG(m, q), especially their relationship with other mathematical structures, we refer to [23, App. B].

Example 5.1.

Fig. 5.1 on the next page represents the well-known projective geometry PG(2, 2) (Fano plane).

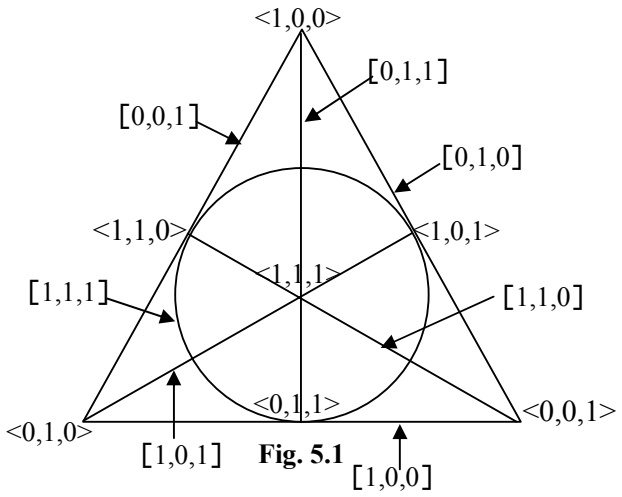

Fig. 5.1

The seven points $<a_0, a_1, a_2>$ correspond to the seven nonzero vectors (a_0, a_1, a_2) of $GF(2)^3$. The seven lines (i.e. one dimensional sub-spaces) $[b_0, b_1, b_2]$ also correspond to the nonzero vectors (b_0, b_1, b_2) of $GF(2)^3$. In fact, the points of incidence with the line $[b_0, b_1, b_2]$ are the three solutions of the equation

$$b_0 x_0 + b_1 x_1 + b_2 x_2 = 0.$$

Since dim PG(2,2) = 2, all these lines are hyperplanes. If we leave out the hyperplane H represented by [1, 0, 0] together with the points it contains, we obtain the structure depicted in Fig. 5.2.

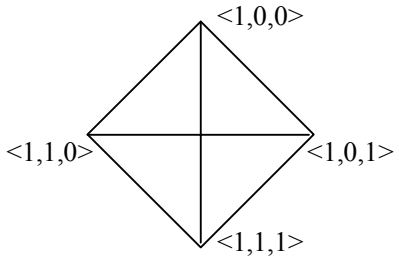

Fig. 5.2

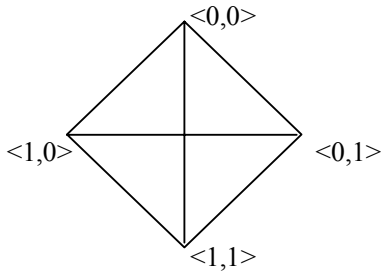

Fig. 5.3

This structure is isomorphic to the Euclidean geometry EG(2,2). More specifically, if we drop the coordinates $a_0 = 1$ in the triples which represent the points, we obtain precisely EG(2,2) itself, since the points of EG(2,2) are by definition the vectors of $GF(2)^2$ (cf. Fig. 5.3).

The Euclidean geometry EG(2,2) contains six one-dimensional subspaces or 1-flats. These are all called EG(1, 2) geometries or briefly EG(1, 2)'s (cf. [23]). There are three parallel systems of 1-flats (three pairs of parallel lines), defined by the linear subspaces

$$S_1 = \{(0,0), (0, 1)\},$$
$$S_2 = \{(0,0), (1, 0)\},$$
$$S_3 = \{(0,0), (1, 1)\}.$$

The parallel systems themselves are

$$\mathcal{P}_1 = \{S_1, S_1 + (1, 0)\},$$
$$\mathcal{P}_2 = \{S_2, S_2 + (0, 1)\},$$
$$\mathcal{P}_3 = \{S_3, S_3 + (0, 1)\}.$$

5.2 Reed-Muller Codes

One way to define Reed-Muller codes (briefly, *RM* codes), is in terms of Euclidean geometries. Since we only consider binary codes, we take $q = 2$ from now on, and consider the geometry EG(m, 2). A subset S of points of EG(m, 2) can be represented by its characteristic vector $\chi(S)$ of length 2^m, containing a 1 in those components which correspond to points of S and 0's elsewhere.

We shall denote the points of EG(m, 2) either by $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{n-1} \in GF(2)^m$, $n = 2^m$, according to the definition of points, or by $P_0, P_1, P_2, \ldots, P_{n-1}$. These conventions imply that we do not distinguish sharply between the vector space $GF(2)^m$ and the geometry EG(m, 2).

Instead of writing $P_0, P_1, P_2, \ldots, P_{n-1}$, we shall occasionally indicate these points by their indices 0, 1, …, $n - 1$. We shall apply this notation especially when dealing with the *support* of a set $S \subseteq$ EG(m, 2). Let S be the set

$$S = \{\mathbf{p}_i, \mathbf{p}_j, \ldots, \mathbf{p}_l\},$$

then we define its support as

$$\sup S = \{i, j, \ldots, l\}.$$

So sup S contains the labels of the positions where $\chi(S)$ has a value 1.

In the remaining part of this thesis, we shall adopt the convention that the vector $\mathbf{p}_i \in GF(2)^m$ corresponding to $P_i \in$ EG(m, 2) is represented by the *reversed* binary representation of the integer i, $0 \le i \le n - 1$, $n = 2^m$. This implies that for any linear subspace S of EG(m, 2), the coset

$$S + \mathbf{p}_a := \{\mathbf{p}_i + \mathbf{p}_a, \mathbf{p}_j + \mathbf{p}_a, \ldots, \mathbf{p}_l + \mathbf{p}_a\} \tag{5.3}$$

can be represented by

$$S \oplus a := \{ i \oplus a, j \oplus a, \ldots, l \oplus a\}, \tag{5.4}$$

where a stands for the m–bit binary representation of the index of $\mathbf{p}$, written in reversed order and where the symbol '$\oplus$' stands for the bitwise *addition modulo* 2 for the length-m vectors $\mathbf{p}_i$, $0 \leq i \leq 2^m - 1$, of $GF(2)^m$, also known as *Nim addition*. The reason why we use the reversed order instead of the normal order (which also implies (5.4)) will be explained in the next few paragraphs. Because of (5.3) and (5.4), we shall denote the coset $S + \mathbf{p}_a$ also by S_a, where we should keep in mind that in general, this index a is not uniquely determined.

A binary code of length $n = 2^m$ can be considered as a set of characteristic vectors $\mathbf{s}$, each of which corresponds to some set $S \subseteq \mathrm{EG}(m, 2)$, i.e. $\mathbf{s} = \chi(S)$. One could also say that such a code is a family of subsets of EG(m, 2). We conclude that any binary vector $\mathbf{v} = (v_0, v_1, \ldots, v_{2^m-1})$, or equivalently, any binary word of length 2^m, describes a subset of EG(m, 2) consisting of those points i (or P_i) for which $v_i = 1$.

It will be obvious that an r-dimensional subspace of EG(m, 2) is represented by a characteristic vector, or codeword, which contains 2^r ones, since there are 2^r points in such a subspace. If the subspace is linear, one of these points is P_0, represented by the m-tuple (0, 0, …, 0), or by the zeroword 00…0 of length m. In particular, a codeword which corresponds to a hyperplane, contains 2^{m-1} ones. If the hyperplane is linear, the point P_0 is one of these. A non-linear hyperplane in EG(m, 2) is the only disjoint coset of a linear hyperplane.

Let H be an arbitrary hyperplane in EG(m, 2). Its characteristic vector $\mathbf{h} = \chi(H)$ contains 2^{m-1} ones and 2^{m-1} zeros. The points $\mathbf{x} = (x_0, x_1, \ldots, x_{m-1}) \in \mathrm{EG}(m, 2)$, which are in H, are determined by some linear equation

$$\mathbf{a}\cdot\mathbf{x} = a_0x_0 + a_1x_1 + \ldots + a_{m-1}x_{m-1} = \delta,$$

where $a_0, a_1, \ldots, a_{m-1}$ and δ are all elements of $GF(2)$. If $\delta = 0$, the hyperplane H is linear, and if $\delta = 1$ it is nonlinear.

Special linear hyperplanes are obtained by taking $\mathbf{a} = \mathbf{e}_j$ (and $\delta = 0$), where $\mathbf{e}_j$ is the j-th (unit) basis vector of the standard basis in $GF(2)^m$, i.e. the vector $\mathbf{e}_j$, $1 \leq j \leq m$, with a one on its $(j - 1)$-th position from the left and zeros elsewhere. (Remember that we label the components p_i of a vector $\mathbf{p} \in GF(2)^m$ from 0 until $m - 1$ and from left to right). We shall denote the hyperplane corresponding to $\mathbf{e}_{m-j+1}$ by H_j, and its characteristic vector by $\mathbf{v}_j = \chi(H_j)$, $1 \leq j \leq m$.

Since we represent P_i as the *reversed* binary representation of $i \in \{0, 1, \ldots, 2^m - 1\}$, it follows that H_m contains the points $P_0, P_1, \ldots, P_{2^{m-1}-1}$, i.e. those points of EG(m, 2) which have an index value less than 2^{m-1}. In general, the linear hyperplane H_j contains those points P_i, the index of which has a binary representation with a zero on the $(m - j)^{\text{th}}$ position from the left.

Example 5.2.

In EG(3,2), the linear hyperplane H_3 is defined by the equation $\mathbf{a}\cdot\mathbf{x} = \mathbf{0}$ with $\mathbf{a} = \mathbf{e}_3 = (0,0,1)$. So, H_3 contains the points (0,0,0), (1, 0, 0), (0,1,0) and (1, 1, 0). According to the above rule, these points are labeled by the binary numbers 000, 001, 010 and 011, respectively. Here, H_3 contains the points P_0, P_1, P_2 and P_3. Similarly, the linear hyperplane H_2 contains the points P_0, P_1, P_4 and P_5 with labels 000, 001, 100 and 101, while the linear hyperplane H_1 contains the points P_0, P_2, P_4 and P_6 with labels 000, 010, 100 and 110.

When writing $H_3 = \{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$, its only coset can be written as

$$H_3 + \mathbf{p}_4 = \{\mathbf{p}_0 + \mathbf{p}_4, \mathbf{p}_1 + \mathbf{p}_4, \mathbf{p}_2 + \mathbf{p}_4, \mathbf{p}_3 + \mathbf{p}_4\},$$
$$= \{\mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7\},$$

(cf. (5.3)). Equivalently, we can write (cf. (5.4))

$$H_3 = \{000, 001, 010, 011\},$$
$$H_3 \oplus 100 = \{100, 101, 110, 111\},$$

When applying structures like these in the next for the construction of snakes, we simply write

$$H_3 = \{0, 1, 2, 3\},$$
$$H_3 \oplus 4 = \{4, 5, 6, 7\}.$$

The parallel system

$$\mathcal{P}_3 = \{\{0, 1, 2, 3\}, \{4, 5, 6, 7\}\}$$

is a cover of EG(3, 2) by hyperplanes. In a similar way, the hyperplanes H_2 and H_1 give rise to covers of EG(3, 2) by hyperplanes, i.e. the parallel systems

$$\mathcal{P}_2 = \{\{0, 1, 4, 5\}, \{2, 3, 6, 7\}\}$$

and

$$\mathcal{P}_1 = \{\{0, 2, 4, 6\}, \{1, 3, 5, 7\}\}$$

are both covers of EG(3, 2).

Let S and T be any pair of subspaces of EG(m, 2) with characteristic vectors $\mathbf{s}$ and $\mathbf{t}$. The *point product* of $\mathbf{s}$ and $\mathbf{t}$ is defined as the vector

$$\mathbf{s}\cdot\mathbf{t} = (s_0t_0, s_1t_1, \ldots, s_{2^{m-1}-1}t_{2^{m-1}-1}),$$

and is the characteristic vector of the intersection $S \cap T$.

In particular, one can consider the point product of two characteristic vectors $\mathbf{v}_i$ and $\mathbf{v}_j$ which represent the linear hyperplanes H_i and H_j, $1 \le i, j \le m$, as defined above. More generally, one can take the point product of r such vectors

$$\mathbf{v} := \mathbf{v}_{i_1} \cdot \mathbf{v}_{i_2} \cdot \ldots \cdot \mathbf{v}_{i_r}, \qquad 0 \le r \le m$$

which represents the linear subspace

$$V := H_{i_1} \cap H_{i_2} \cap \ldots \cap H_{i_r}.$$

This point product is called a product of degree r. Special cases are $r = 0$ with $\mathbf{v} = \mathbf{1}$, representing the whole space EG(m, 2), and $r = m$ yielding $\mathbf{v} = 100\ldots0$ which corresponds to

the zero space {**0**} of EG(m, 2). A linear combination of such products with coefficients from $GF(2)$ is called a *polynomial* in $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ of degree r, if r is the highest degree which occurs. Here, $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ are independent vectors corresponding to 'independent' hyperplanes.

Definition 5.2

The binary r-th order Reed-Muller code $R(r, m)$ of length n, is the linear code consisting of all polynomials in $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ of degree at most r, where $n = 2^m$.

One could also say that the vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$, representing the special hyperplanes $H_1, H_2, \ldots, H_m$ of EG(m, 2), *generate* the code $R(r, m)$, be it that they do not constitute a basis in general.

Example 5.3.

In Example 5.2, we introduced EG(3,2) by using the 'binary' labeling of the points, the binary hyperplanes $H_3 = \{0, 1, 2, 3\}$, $H_2 = \{0, 1, 4, 5\}$ and $H_1 = \{0, 2, 4, 6\}$. The characteristic vectors of these hyperplanes are, respectively,

$$\mathbf{v}_3 = 11110000,$$
$$\mathbf{v}_2 = 11001100,$$
$$\mathbf{v}_1 = 10101010.$$

The code $R(1, 3)$ of length $2^3 = 8$ is the linear code generated by **1** and the vectors $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$ (cf. Fig. 5.4 in the next section). Since the four generating vectors are independent, the dimension of $R(1, 3)$ is equal to 4, which will be confirmed by Theorem 5.3.

For the code $R(2, 3)$ we also need, in addition to the vectors **1**, $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, the products of degree 2

$$\mathbf{v}_3 \cdot \mathbf{v}_1 = 10100000,$$
$$\mathbf{v}_3 \cdot \mathbf{v}_2 = 11000000,$$
$$\mathbf{v}_2 \cdot \mathbf{v}_1 = 10001000.$$

The seven vectors are independent and generate a linear [8, 7, 2]-code consisting of all even-weight binary words of length 8.

We conclude this section by presenting some well-known properties of *RM*-codes (cf. [23, Chapter 13]) which play a major role in the next.

Theorem 5.3

For any m and r, $0 \leq r \leq m$, the code $R(r,m)$ is an $[n, k, d]$-code with parameter values

$$n = 2^m, \qquad k = \sum_{i=0}^{r} \binom{m}{i}, \qquad d = 2^{m-r}. \tag{5.5}$$

Let $m > 0$, $n = 2^m$ and let the notation $G(r, m)$ stand for a matrix the rows of which are independent basis vectors that generate $R(r, m)$, for $0 \leq r \leq m$. Notice that $R(m, m)$ is the whole

space $GF(2)^n$. In order to define a general formula for the matrix $G(r, m)$, we define the two special matrices $G(0, m) = (\mathbf{1})$ being the matrix consisting of the all-1 row vector of length $n = 2^m$ and $G(m, m)$, which is the identity matrix I_n.

Using the same integer k of (5.5), the following recursive formula provides us with a basis of $R(r + 1, m + 1)$ (cf. [23] Theorem 13.2 and [14] Sect. 1.10).

Theorem 5.4

For every r with $0 \le r < m$, where $n = 2^m$, we have

$$G(r+1, m+1) = \begin{pmatrix} G(r+1,m) & G(r+1,m) \\ 0 & G(r,m) \end{pmatrix}, \tag{5.6}$$

where 0 is the $k \times n$ zero matrix.

Example 5.4.

According to Theorem 5.4, we construct the following bases

$$G(1,2) = \begin{pmatrix} G(1,1) & G(1,1) \\ 00 & G(0,1) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix},$$

and

$$G(1,3) = \begin{pmatrix} G(1,2) & G(1,2) \\ 0000 & G(0,2) \end{pmatrix} = \begin{pmatrix} 10101010 \\ 01010101 \\ 00110011 \\ 00001111 \end{pmatrix}.$$

One can immediately verify that the above basis of $R(1, 3)$ is equivalent to the one constructed in Example 5.3.

Theorem 5.5

The codewords of minimum weight in $R(r,m)$ are precisely the $(m - r)$-flats in EG$(m, 2)$

Theorem 5.6

The code $R(r,m)$ is spanned by its minimum-weight codewords.

For the proofs, we again refer to [23].

5.3. Parallel Systems in EG(3,2) and EG(4,2)

Our first example is the Reed-Muller code $R(1,3)$ in terms of the geometry EG(3,2). We first repeat some facts which were discussed already in Examples 5.2 and 5.3. The points of this geometry are the 2^3 triples $\mathbf{x} = (x_1, x_2, x_3) \in GF(2)^3$. According to the conventions in Section 5.2, Example 5.2, we denote these points by $P_0, P_1, \ldots, P_7$, or by 0, 1, …, 7, which stand for the binary words 000, 001, …, 111. The following list consists of the 16 codewords of $R(1,3)$ (cf.Table 13.1 in [23]).

	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1
$\mathbf{v}_3$	1	1	1	1	0	0	0	0
$\mathbf{v}_2$	1	1	0	0	1	1	0	0
$\mathbf{v}_1$	1	0	1	0	1	0	1	0
$\mathbf{v}_2 + \mathbf{v}_3$	0	0	1	1	1	1	0	0
$\mathbf{v}_1 + \mathbf{v}_3$	0	1	0	1	1	0	1	0
$\mathbf{v}_1 + \mathbf{v}_2$	0	1	1	0	0	1	1	0
$\mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$	1	0	0	1	0	1	1	0
$\mathbf{1} + \mathbf{v}_3$	0	0	0	0	1	1	1	1
$\mathbf{1} + \mathbf{v}_2$	0	0	1	1	0	0	1	1
$\mathbf{1} + \mathbf{v}_1$	0	1	0	1	0	1	0	1
$\mathbf{1} + \mathbf{v}_2 + \mathbf{v}_3$	1	1	0	0	0	0	1	1
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_3$	1	0	1	0	0	1	0	1
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_2$	1	0	0	1	1	0	0	1
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$	0	1	1	0	1	0	0	1

Fig. 5.4

The 16 words of $R(1,3)$ correspond to the 16 polynomials of degree ≤ 1 in $\mathbf{v}_3$, $\mathbf{v}_2$ and $\mathbf{v}_1$ defined by the equation $\mathbf{e}_i \cdot \mathbf{x} = 0$, for $i = 3$, 2 and 1, respectively. These vectors represent the following linear hyperplanes in EG(3,2)

$$H_3 = \{0, 1, 2, 3\} = \{000, 001, 010, 011\},$$
$$H_2 = \{0, 1, 4, 5\} = \{000, 001, 100, 101\},$$
$$H_1 = \{0, 2, 4, 6\} = \{000, 010, 100, 110\}.$$

One can immediately verify that these sets are closed w.r.t. the ⊕-operation (cf. (5.4)), as they should be since they are linear spaces.

From Theorem 5.3, we know that the dimension of $R(1, 3)$ is 4, and hence we can take as basis the four independent words **1**, $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$. Therefore, the general codeword of this code can be written as

$$a_0\mathbf{1} + a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + a_3\mathbf{v}_3,$$

with $a_i \in GF(2)$. We can also choose a minimum-weight basis according to Theorem 5.6, e.g. $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{1} + \mathbf{v}_1\}$.

The linear hyperplane H_3 has one coset which can be written (cf. (5.4)) e.g. as $H_3 \oplus 4$. Together, H_3 and $H_3 \oplus 4$ cover EG(3, 2), and hence $\mathcal{P}_3 = \{H_3, H_3 \oplus 4\}$ is a parallel system of 2-flats of this geometry (cf. Section 5.1). Similarly, $\mathcal{P}_1 = \{H_1, H_1 \oplus 1\}$ and $\mathcal{P}_2 = \{ H_2, H_2 \oplus 2\}$ are parallel systems of EG(3,2).

Similarly, we can construct parallel systems of 1-flats of EG(3,2). Since a 1-flat is a line, it is determined by two points of EG(3,2). E.g. the pair of points {0, 1} constitutes a line which passes through the origin, so $L_1 = \{0,1\}$ is a linear 1-dimensional subspace of EG(3,2). Its cosets $L_2 = \{2, 3\}$, $L_3 = \{4, 5\}$ and $L_4 = \{6,7\}$ are non-linear 1-dimensional subspaces. Altogether, the family $\mathcal{P} = \{\{0,1\}, \{2,3\}, \{4,5\}, \{6,7\}\}$ is a parallel system of 1-flats,

partitioning the points of EG(3,2) in 4 classes of 2 points.

Our second example is the Reed-Muller code $R(2, 4)$, based on EG(4, 2). The points of EG(4,2) are the 2^4 quadruples $\mathbf{x} = (x_1, x_2, x_3, x_4) \in GF(2)^4$. The code is generated by the characteristic vectors of the hyperplanes

$$H_4 = \{0, 1, 2, 3, 4, 5, 6, 7\},$$
$$H_3 = \{0, 1, 2, 3, 8, 9, 10, 11\},$$
$$H_2 = \{0, 1, 4, 5, 8, 9, 12, 13\},$$
$$H_1 = \{0, 2, 4, 6, 8, 10, 12, 14\}.$$

To obtain all codewords, one has to take all polynomials of these characteristic vectors of degree 2 or less. From Theorem 5.3, it follows that the dimension of $R(2, 4)$ is equal to 11.

Fig. 5.5 shows the 11 spanning vectors of $R(2, 4)$ which correspond to point products of $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ and $\mathbf{v}_4$ of degree 0, 1 and 2. Each codeword of $R(2,4)$ is a linear combination of these vectors, and hence it corresponds to a polynomial in $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ and $\mathbf{v}_4$ of degree at most 2.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$\mathbf{v}_4$	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
$\mathbf{v}_3$	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
$\mathbf{v}_2$	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
$\mathbf{v}_1$	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
$\mathbf{v}_3 \cdot \mathbf{v}_4$	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
$\mathbf{v}_2 \cdot \mathbf{v}_4$	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
$\mathbf{v}_1 \cdot \mathbf{v}_4$	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0
$\mathbf{v}_2 \cdot \mathbf{v}_3$	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0
$\mathbf{v}_1 \cdot \mathbf{v}_3$	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0
$\mathbf{v}_1 \cdot \mathbf{v}_2$	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0

Fig. 5.5

From Theorem 5.3 and 5.4, it follows that $R(2, 4)$ is a [16, 11, 4]-code which can also be spanned by a minimum-weight basis, as we shall see in the next section.

5.4. Snakes Embedded in EG(3,2)

In this section, we discuss a relationship between the theory of Euclidean geometries as touched upon in Section 5.3, and the method to construct snake-in-the-box codes which was introduced in Section 4.1 of this thesis.

Our starting point is the parallel system of lines (cf. (5.4))

$$\mathcal{P}_1 = \{L_1, L_2, L_3, L_4\} = \{\{0, 1\}, \{2, 3\}, \{4, 5\}, \{6, 7\}\}, \tag{5.7}$$

which partitions the set of points of EG(3,2) in four equivalence classes of size 2. Next, we consider a set $\mathcal{B}$ of 2-flats which intersect each of the four lines in precisely one point. As usual, we identify these 2-flats with their supports, as we did with the 1-flats in (5.7) above. Because of the role they are going to play in the next, we call such 2-flats *blocks*, and instead of the parentheses usually used to denote an unordered set, we shall write round brackets to

indicate that the integers of the set are ordered from left to right. So, the block

$$B_i = (i_1, i_2, i_3, i_4)$$

represents a 2-flat of EG(3, 2) containing the points P_{i_1}, P_{i_2}, P_{i_3} and P_{i_4}. We also speak of an *ordered block* (cf. [34-39])

The integers in B_i are arranged such that i_j is the point of intersection with the line L_j, $1 \leq j \leq 4$. Hence, we can say that each integer of the set $\{0, 1, \ldots, 7\}$ has a fixed position in any of the blocks of $\mathcal{B}$, in which it occurs. In Chapter 4, we called such a family $\mathcal{B}$ a *block list* which satisfies the *fixed-position property*, and we said that the blocks of $\mathcal{B}$ have a *fixed internal order*. In general, we are interested in block lists the blocks of which correspond to the vectors of a minimum-weight-d basis spanning a linear $[n, k, d]$-code. Here, we take $d = 4$.

It will appear in the next that a Reed-Muller code of type $R(m - 2, m)$ is an appropriate code to yield such a block list. From Theorem 5.3, we know that $R(m - 2, m)$ has minimum weight 4, and according to Theorem 5.5, we can always find a minimum-weight basis. The only thing yet to find out is whether we can select a minimum-weight basis the corresponding blocks of which satisfy the fixed-position property.

The following list shows *all* blocks of EG(3,2) which intersect each of the lines of $\mathcal{P}$ in precisely one point.

	L_1	L_2	L_3	L_4
$\mathbf{v}_1$	(0,	2,	4,	6)
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_3$	(0,	2,	5,	7)
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_2$	(0,	3,	4,	7)
$\mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$	(0,	3	5,	6)
$\mathbf{1} + \mathbf{v}_1$	(1,	3,	5,	7)
$\mathbf{v}_1 + \mathbf{v}_3$	(1,	3,	4,	6)
$\mathbf{v}_1 + \mathbf{v}_2$	(1,	2	5,	6)
$\mathbf{1} + \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$	(1,	2,	4,	7)

Fig. 5.6

We remark that the total number of 2-flats in EG(3,2) equals $2^{3-2}\begin{bmatrix} 3 \\ 2 \end{bmatrix} = 14$, according to Theorem 5.1. Hence, there are 6 blocks which do not satisfy the fixed-position property. It will be obvious that if a block satisfies the fixed-position property, then its only coset does, and vice versa. Fig. 5.7 below shows the set of all blocks partitioned in pairs of blocks which are cosets of each other. The first four pairs satisfy the fixed-position property with respect to the parallel system (5.7).

$$\begin{array}{ll} B_1 = (0, 2, 4, 6), & B_1 \oplus 1 = (1, 3, 5, 7) \\ B_2 = (0, 2, 5, 7), & B_2 \oplus 1 = (1, 3, 4, 6) \\ B_3 = (0, 3, 4, 7), & B_3 \oplus 1 = (1, 2, 5, 6) \\ B_4 = (0, 3, 5, 6), & B_4 \oplus 1 = (1, 2, 4, 7) \end{array}$$

$B_5 = (0, 1, 2, 3)$, $B_5 \oplus 6 = (4, 5, 6, 7)$
$B_6 = (0, 1, 4, 5)$, $B_6 \oplus 2 = (2, 3, 6, 7)$
$B_7 = (0, 1, 6, 7)$, $B_7 \oplus 4 = (2, 3, 4, 5)$

Fig. 5.7

It appears that we can select a set of four different blocks satisfying the fixed-position property, such that the corresponding vectors are independent, and which therefore constitute a minimum-weight basis of $R(1, 3)$. We shall speak of a set of *independent blocks*. Two such sets of independent blocks satisfying the fixed-position property are:

$B_1 = (0, 3, 4, 7)$, $B'_1 = (1, 3, 5, 7)$,
$B_2 = (1, 2, 5, 6)$, $B'_2 = (0, 2, 4, 6)$,
$B_3 = (1, 3, 4, 6)$, $B'_3 = (0, 3, 5, 6)$,
$B_4 = (1, 3, 5, 7)$. $B'_4 = (0, 3, 4, 7)$,

Fig. 5.8

where we relabeled or renamed the blocks chosen from Fig. 5.7. This labeling is such that the (external) order of the blocks meets the conditions of "well-ordered" (cf. Definition 4.3 in Section 4.2).

From Corollary 4.3, it now follows that the first list

$$\mathcal{B} = (B_1, B_2, B_3, B_4) \tag{5.8}$$

generates a symmetric snake-in-the-box code (snake) by applying transition sequence (4.8) with $k = 4$. The complete sequence is

$\overline{S}_4(\mathcal{B}) = B_1 B_2 B_1 B_3 B_1 B_2 B_1 B_4 B_1 B_2 B_1 B_3 B_1 B_2 B_1 B_4$

= 0, 3, 4, 7, 1, 2, 5, 6, 0, 3, 4, 7, 1, 3, 4, 6, 0, 3, 4, 7, 1, 2, 5, 6, 0, 3, 4, 7, 1, 3, 5, 7,
0, 3, 4, 7, 1, 2, 5, 6, 0, 3, 4, 7, 1, 3, 4, 6, 0, 3, 4, 7, 1, 2, 5, 6, 0, 3, 4, 7, 1, 3, 5, 7.

In the table below, the list of codewords of $R(1,3)$ is presented.

1. 00000000, 9. 11110000,
2. 10011001, 10. 01101001,
3. 11111111, 11. 00001111,
4. 01100110, 12. 10010110,
5. 00111100, 13. 11001100,
6. 10100101, 14. 01010101,
7. 11000011, 15. 00110011,
8. 01011010, 16. 10101010,

Fig. 5.9

For the complete list of the words that constitute the snake itself, we refer to Appendix B. The table in Fig. 5.9 shows clearly that the list of the codewords of $R(1, 3)$, and hence also of the snake, is symmetric, since the second half is obtained from the first half by translating this half over the vector 11110000.

Notice that the two lists in Fig.5.8 produce distinct but equivalent snakes which can be transferred into each other by relabeling or renaming the integers. Other choices of four independent blocks from the set of blocks in Fig. 5.7 will give rise to different snakes.

However, all such snakes are based on the parallel system $\mathcal{P}$ defined in (5.7).

Let us consider in more detail the snake generated by the block list

$$\mathcal{B}' = (B'_1, B'_2, B'_3, B'_4) \tag{5.9}$$

from Fig.5.8. Just like (5.8), this block list generates a snake in Q_8 of length $2^6 = 4 \cdot 2^4$. The binary code generated by the basis vectors $\mathbf{b}'_1$, $\mathbf{b}'_2$, $\mathbf{b}'_3$ and $\mathbf{b}'_4$ which correspond to the blocks in (5.9) is an [8, 4, 4]-code (the Reed-Muller code $R(1, 3)$), which is identical to the extended Hamming code H_3^{ext}.

By removing block B'_1 from the block system (5.9), we are left with a smaller system which of course is still well-ordered. By Corollary 4.3 this system generates a snake of length 2^5, while the underlying code is an [8, 3, 4]-code. However, the effective length has decreased by 1 due to the fact that B'_1 is the only block in (5.9) containing the integer 1. Therefore, although the snake is in Q_8, it actually is in a subgraph of Q_8 which is isomorphic to Q_7.

If we omit the bit position 1 from the codewords (*puncturing* the code), we obtain a snake-in-the-box of length 2^5 in Q_7, and the underlying linear code has parameters [7, 3, 4]. We can continue by omitting block B'_4 which is the only block containing the integer 7. After puncturing with respect to bit position 7, we obtain a snake in Q_6 generated by $B'_2 = (0, 2, 4, 6)$ and $B'_3 = (0, 3, 5, 6)$, which is of length 2^4, while the underlying code is a [6, 2, 4]-code.

Finally, we can omit one of the two remaining blocks, say B'_3. Our method introduced in Chapter 4 now provides us with a snake in Q_6 with transition sequence

$$\overline{S}_1(B'_2) = 0, 2, 4, 6, 0, 2, 4, 6.$$

Since the removed block B'_3 contains two integers 3 and 5, which do not occur in the remaining block system (B'_2), we now can puncture with respect to these two positions. So, the resulting snake of length 2^3 not only is situated in a subgraph of Q_6 isomorphic to Q_5, but it also can be considered as a snake of length 2^3 in Q_4 (by puncturing twice). The underlying codes of the snakes in Q_5 and in Q_4 are a [5, 1, 4]-code and a [4, 1, 4]-code, respectively. The latter code is the Reed-Muller code $R(0, 2)$, which is identical to the extended Hamming code H_2^{ext}.

5.5 Snakes Embedded in EG(4, 2)

Next, we shall construct snakes embedded in EG(4,2). In EG(4, 2), we introduce a parallel system of 2-dimensional subspaces (2-flats). Since such a subspace is isomorphic to the geometry EG(2,2) we call them planes. Similar to the parallel system of lines in EG(3, 2), we define

$$\begin{aligned}\mathcal{P} &= \{V_1, V_2, V_3, V_4\} \\ &= \{\{0, 1, 2, 3\}, \{4, 5, 6, 7\}, \{8, 9, 10, 11\}, \{12, 13, 14, 15\}\}.\end{aligned} \tag{5.10}$$

We remind the reader that V_1 is the only linear subspace of EG(4,2) in $\mathcal{P}$, whereas V_2, V_3 and

V_4 are cosets of V_1.

One can easily verify that there are 16 linear 2-flats which intersect each of the subspaces V_1, V_2, V_3 and V_4 in precisely one point. These 2-flats - written as ordered blocks - are presented by the following list.

	V_1	V_2	V_3	V_4
B_1	0,	4,	8,	12)
B_2	0,	4,	10,	14)
B_3	0,	4,	11,	15)
B_4	0,	4,	9,	13)
B_5	0,	5,	8,	13)
B_6	0,	5,	9,	12)
B_7	0,	5,	10,	15)
B_8	0,	5,	11,	14)
B_9	0,	6,	8,	14)
B_{10}	0,	6,	9,	15)
B_{11}	0,	6,	10,	12)
B_{12}	0,	6,	11,	13)
B_{13}	0,	7,	8,	15)
B_{14}	0,	7,	9,	14)
B_{15}	0,	7,	10,	13)
B_{16}	0,	7,	11,	12)

Fig. 5.10

Each of the subspaces B_i has three disjoint cosets $B_i \oplus 1$, $B_i \oplus 2$, $B_i \oplus 3$ which also intersect each of the V_j, $j \in \{1, 2, 3, 4\}$, in precisely one point. So, altogether we have 64 blocks which satisfy the fixed-position property. In order to construct a snake based on an appropriate basis of $R(2, 4)$, we have to select 11 of these 64 blocks, which are independent and which can be well-ordered in the sense of the previous section.

The following procedure to find 11 independent blocks seems 'obvious'. First we select 8 independent blocks from the list in Fig. 5.10, and next we take 3 cosets of one of these blocks. Along these lines, we obtain e.g. the list of independent blocks in Fig. 5.11.

$$
\begin{aligned}
B_1 &= (0, 4, 8, 12)\\
B_5 &= (0, 5, 8, 13)\\
B_9 &= (0, 6, 8, 14)\\
B_{11} &= (0, 6, 10, 12)\\
B_{13} &= (0, 7, 8, 15)\\
B_{14} &= (0, 7, 9, 14)\\
B_{15} &= (0, 7, 10, 13)\\
B_{16} &= (0, 7, 11, 12)\\
B_1 \oplus 1 &= (1, 5, 9, 13)\\
B_1 \oplus 2 &= (2, 6, 10, 14)\\
B_3 \oplus 3 &= (3, 7, 11, 15).
\end{aligned}
$$

Fig. 5.11

By rearranging these blocks and by relabeling their indices, we obtain the list $\mathcal{B}$ presented in

Fig. 5.12.

B_1 = (3, 7, 11, 15),
B_2 = (2, 6, 10, 14),
B_3 = (1, 5, 9, 13),
B_4 = (0, 4, 8, 12),
B_5 = (0, 5, 8, 13),
B_6 = (0, 6, 8, 14).
B_7 = (0, 7, 8, 15),
B_8 = (0, 7, 9, 14),
B_9 = (0, 7, 10, 13),
B_{10} = (0, 7, 11, 12),
B_{11} = (0, 6, 10, 12),

Fig. 5.12

The sublist B_1, B_2, …, B_{10} satisfies the condition for being well-ordered (after having permuted the integers 0, 1, …, 15 in an appropriate way). However, when adding B_{11}, the list is not well-ordered anymore. The integer 6_2 (= 6), which is on the second position in block B_6, now also occurs in B_{11}. Since 6_4 (= 14) occurs in B_8, both integers 6_2 and 6_4 occur in blocks B_l, $l > 6$, and hence condition (*iii*) of the definition of a well-ordered block system is not satisfied.

A computer check shows that the ordered code generated by transition sequence (4.8), based on the block system of Fig. 5.12**,** is not a snake. In fact, the 3327-th word 0000001101011000 and the 4094-th word 0000001100011000 differ at the 10-th position. Another pair of words at distance 1 is the (4096 + 3327) = 7423-th word $\mathbf{b}_{10} + \mathbf{b}_{11} + \mathbf{w}_{3347}$ = 0000000001101000 and the (4096 + 4094) = 8190-th word $\mathbf{b}_{10} + \mathbf{b}_{11} + \mathbf{w}_{8090}$ = 0000000000101000. This of course, is due to the symmetry of the transition sequence.

Because of the above considerations, it will be clear that by omitting block B_{11}, we obtain a list which is well-ordered, and hence, by Corollary 4.3, generates a snake of length $4 \cdot 2^{10} = 2^{12}$ in Q_{16}. However, we prefer to have a snake of length 2^{13} in Q_{16}.

We did not succeed, by permuting the blocks B_1, B_2, …, B_{11}, in finding a block system which does satisfy the "well-ordering" condition. However, certain permutations of the blocks, although not well-ordered, give rise to block lists which *do* generate a snake, as was verified by a computer program. We found this to be the case e.g. for the following block list.

B_1 = (3, 7, 11, 15),
B_2 = (2, 6, 10, 14),
B_3 = (1, 5, 9, 13),
B_4 = (0, 4, 8, 12),
B_5 = (0, 6, 10, 12),
B_6 = (0, 7, 8, 15),
B_7 = (0, 5, 8, 13),
B_8 = (0, 7, 10, 13),
B_9 = (0, 7, 11, 12),
B_{10} = (0, 7, 9, 14),
B_{11} = (0, 6, 8, 14).

Fig. 5.13

It turned out, by applying a computer program, that if one substitutes $k = 11$ and block B_i from Fig. 5.13 for each integer i, $1 \leq i \leq 11$, in sequence (4.8), the result is a transition sequence for a snake of length $4.2^{11} = 2^{13}$. This example shows that the condition of being well-ordered for a block list is sufficient to construct a snake, but not *necessary*. That the list of Fig. 5.13 indeed generates a snake can also be proved by verifying that the conditions of Theorem 4.2 hold. These conditions are not only sufficient, but necessary as well.

Theorem 5.7

*The block list of Fig.*5.13 *generates a snake in* Q_{16} *of length* 2^{13}.

Proof. If one leaves out block B_5 from the list in Fig. 5.13, the remaining list is well-ordered. So, we only have to verify conditions $(i) - (iv)$ of Theorem 4.2 for $i = 6$. The only codewords $\mathbf{c}$ of weight 4 which can be written as (4.13) with $|X|$ odd, are the codewords with support

$$\{0, 6, 10, 12\}$$

and

$$\{0, 6, 11, 13\},$$

which are not of one of the types $(i) - (iv)$. □

An alternative proof can be given by applying Theorem 4.4, since the list of Fig.5.13 is lexicographically ordered.

Next, we want to construct a snake of length 2^{12} in Q_{15}. In order to obtain such a snake, we could remove a block from the block list in Fig.5.13 containing precisely one integer which does not occur in the other blocks. Blocks B_1, B_2, B_3 and B_4 all satisfy this condition. However, when removing one of these blocks, the remaining list of 10 blocks is not well-ordered, and so we can not apply Corollary 4.3. Therefore, we change the list of Fig.5.12 in the following way.

Block B_1 of Fig. 5.12 is replaced by $B_1 \Delta B_7 \Delta B_{10}$ giving the list in Fig. 5.14, where we also relabeled the blocks. The notation $A \Delta B := \{x \mid x \in A \text{ or } x \in B, x \notin A \cap B\}$, stands for the symmetric difference of the sets A and B.

$$\begin{aligned}
B_1 &= (3, 7, 8, 12),\\
B_2 &= (2, 6, 10, 14),\\
B_3 &= (1, 5, 9, 13),\\
B_4 &= (0, 4, 8, 12),\\
B_5 &= (0, 5, 8, 13),\\
B_6 &= (0, 6, 8, 14).\\
B_7 &= (0, 7, 8, 15),\\
B_8 &= (0, 7, 9, 14),\\
B_9 &= (0, 7, 10, 13),\\
B_{10} &= (0, 7, 11, 12),\\
B_{11} &= (0, 6, 10, 12),
\end{aligned}$$

Fig. 5.14

Since now B_7 is the only block that contains 15, we obtain a block list that corresponds to a linear [15, 10, 4]-code when omitting this block. However, the remaining blocks are not well-ordered, since the integers 6_2 (= 6) and 6_4 (= 14) both occur in blocks B_l, $l > 6$.

We now change the internal order in the blocks by interchanging the integers i_3 and i_4 for all i. Moreover, we interchange blocks B_9 and B_{10}. Omitting now block B_7 provides us with the list of Fig. 5.15.

$B_1 =$ (3, 7, 12, 8),
$B_2 =$ (2, 6, 14, 10),
$B_3 =$ (1, 5, 13, 9),
$B_4 =$ (0, 4, 12, 8),
$B_5 =$ (0, 5, 13, 8),
$B_6 =$ (0, 6, 14, 8).
$B_7 =$ (0, 7, 14, 9),
$B_8 =$ (0, 7, 12, 11),
$B_9 =$ (0, 7, 13, 10),
$B_{10} =$ (0, 6, 12, 10),

Fig. 5.15

This list is well-ordered and hence it generates a snake of length 2^{12} in Q_{15}.

In a similar way as in Section 5.4, we now can reduce the list in Fig. 5.15 to get snakes in smaller hypercubes. Successively, we remove blocks B_1, B_2, B_3, B_4, B_5 and B_6. The remaining lists give rise to snakes in Q_i of length 2^{i-3}, for i = 14, 13, …, 9, respectively, after puncturing with respect to the integers 3, 2, 1, 4, 5, 8. The underlying codes of the snakes are [14, 9, 4]-, [13, 8, 4]-, …, [8, 4, 4]-codes, respectively. We are left now with the list (B_7, B_8, B_9, B_{10}).

If we remove block B_9 and puncture with respect to the integer 13, our construction gives a snake of length 2^5 in Q_8. However, we can do better by removing B_7 instead of B_9, since B_7 contains two integers, 14 and 9, which are not present in the other three blocks. Proceeding like this gives a snake of length 2^5 in Q_7 which has the Hamming [7, 3, 4]-code as its underlying linear code. Such a snake was also constructed in Section 5.4 when we started from the Reed-Muller code $R(1, 3)$.

An alternative way to obtain a snake in Q_{15} of length 2^{12} is to remove B_1 from the list of Fig. 5.13, as suggested already on the previous page, and to apply Theorem 4.4. Similarly, we obtain snakes in Q_i, i = 14, 13, …, 9, by removing B_2, B_3, B_4, B_6, B_7, B_9, respectively, every time applying Theorem 4.4.

As we noticed already, the snake of length 2^{13} in Q_{16} generated by the list of Fig. 5.13, was first found by computer search. Only then it was verified being a snake by applying Theorem 2.2. However, we can also use this theorem for a straightforward construction of snakes, especially when a well-ordered sublist is at hand. We shall illustrate this by the following examples.

Example 5.5

We rearrange the blocks of the list in Fig. 5.13.

$B_1 = (\ 3, 7, 11, 15),$
$B_2 = (\ 2, 6, 10, 14),$
$B_3 = (\ 1, 5,\ \ 9, 13),$
$B_4 = (\ 0, 4,\ \ 8, 12),$
$B_5 = (\ 0, 6, 10, 12).$
$B_6 = (\ 0, 7,\ \ 8, 15),$
$B_7 = (\ 0, 7,\ \ 9, 14),$
$B_8 = (\ 0, 5,\ \ 8, 13),$
$B_9 = (\ 0, 7, 10, 13),$
$B_{10} = (\ 0, 7, 11, 12).$
$B_{11} = (\ 0, 6,\ \ 8, 14).$

Fig. 5.16

The above list is still lexicographically ordered with respect to the integers of I_1. So we can apply Theorem 4.4. One can immediately verify that $\mathbf{c} = \mathbf{b}_{i-1}$ does not meet the conditions (*i*) and (*ii*) of that Theorem, since if $(i-1)_2 = i_2$ then $(i-1)_4 \neq j_4$ for $j > i$ and if $(i-1)_4 = i_4$ then $(i-1)_2 \neq j_2$ for $j > i$. Hence, a codeword (cf. (4.13))

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

as mentioned in the Theorem, has to consist of at least three basis vectors if it is of type (*i*) or (*ii*). From the structure of the list in Fig. 5.16, it is clear that for such a vector **c** we must have $i = 6$ and that $W = sup\ \mathbf{c}$ has to be of the form {0, 7, …, …}, {0, 6, …, …} or {0, 5, …, …}.

However, if $W = \{0, 7, \ldots, \ldots\}$, it would follow that $\mathbf{c} = \mathbf{b}_l$ for $l \in \{6, 7, 9, 10\}$, which contradicts our earlier conclusion w.r.t. **c**. If $W = \{0, 6, \ldots, \ldots\}$, the only possibilities are $W = B_6 \Delta B_7 \Delta B_{11} = \{0, 6, 9, 15\}$ and $W = B_5 \Delta B_9 \Delta B_{10} = \{0, 6, 11, 13\}$. In the first case, we would have $i = 7$, but $7_2 \neq 6$ and $7_4 \neq 15$, while in the second case $i = 6$, but $6_2 \neq 6$ and $6_4 \neq 13$.

If $W = \{0, 5, \ldots, \ldots\}$, the only possibilities are $W = \{0, 5, 10, 15\}$, $W = \{0, 5, 9, 12\}$, and $W = \{0, 5, 11, 14\}$. We can write $B_6 \Delta B_8 \Delta B_9 = \{0, 5, 10, 15\}$, yielding $i = 7$, but $7_2 \neq 5$, $7_4 \neq 15$. Similarly, $B_5 \Delta B_7 \Delta B_8 \Delta B_9 \Delta B_{11} = \{0, 5, 9, 12\}$ yielding $i = 6$, but $6_2 \neq 5$, $6_4 \neq 12$. Finally, $B_5 \Delta B_8 \Delta B_9 \Delta B_{10} \Delta B_{11} = \{0, 5, 11, 14\}$ yields $i = 6$, but $6_2 \neq 5$, $6_4 \neq 14$.

So in all cases we proved that $\mathcal{C}$ does not contain words of type (*i*) or (*ii*) of Theorem 4.4, and we may conclude that the list in Fig.5.16 generates a snake $\mathcal{S}$.

Example 5.6

Starting from the list of Fig. 5.16, we shall now try to obtain permutations of this list which also generate a snake when we apply our construction. We restrict ourselves to permutations of the blocks B_4 until B_{11}. According to Theorem 4.4, in order to get a snake, these blocks must not generate a support W corresponding to a codeword $\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$ such that

(*i*) $W = \{0, i_2, p_3, j_4\}$, or

(*ii*) $W = \{0, j_2, p_3, i_4\}$.

In addition to the basis blocks themselves, we have the following supports which are of the above type:

a. $\{0, 6, 11, 13\} = \{0, 6, 10, 12\} \Delta \{0, 7, 11, 12\} \Delta \{0, 7, 10, 13\}$;

b. $\{0, 6, 9, 15\} = \{0, 6, 8, 14\} \Delta \{0, 7, 9, 14\} \Delta \{0, 7, 8, 15\}$;

c. $\{0, 5, 10, 15\} = \{0, 5, 8, 13\} \Delta \{0, 7, 10, 13\} \Delta \{0, 7, 8, 15\}$;

d. $\{0, 5, 11, 14\} = \{0, 5, 8, 13\} \Delta \{0, 6, 10, 12\} \Delta \{0, 6, 8, 14\} \Delta \{0, 7, 10, 13\}$ $\Delta \{0, 7, 11, 12\}$;

e. $\{0, 5, 9, 12\} = \{0, 5, 8, 13\} \Delta \{0, 6, 10, 12\} \Delta \{0, 6, 8, 14\} \Delta \{0, 7, 9, 14\}$ $\Delta \{0, 7, 10, 13\}$,

f. $\{0, 4, 9, 13\} = \{0, 4, 8, 12\} \Delta \{0, 6, 8, 14\} \Delta \{0, 6, 10, 12\} \Delta \{0, 7, 9, 14\}$ $\Delta \{0, 7, 10, 13\}$,

g. $\{0, 4, 10, 14\} = \{0, 4, 8, 12\} \Delta \{0, 6, 8, 14\} \Delta \{0, 6, 10, 12\}$

h. $\{0, 4, 11, 15\} = \{0, 4, 8, 12\} \Delta \{0, 7, 8, 15\} \Delta \{0, 7, 11, 12\}$.

We build up a block list in the following heuristic way.

Let us start e.g. by defining $B_4 = \{0, 6, 10, 12\}$. Next we choose $B_5 = \{0, 7, 8, 15\}$. This choice guarantees that there is no support W of type (*i*) or (*ii*) which contains B_4 in its block decomposition, i.e. supports *a*, *d*, *e*, *f* and *g* as well as B_4 itself will not give rise to a violation of the conditions of Theorem 4.4. The reason is that B_5 does neither contain the second nor the fourth integer of any of these supports. Our next choice is $B_6 = \{0, 6, 8, 14\}$, which eliminates the supports B_5, *b* and *c*. Though the second integer of support *b*, i.e. 6, occurs in B_6, it still satisfies the condition of Theorem 4.4 since 15 – its fourth integer – does not occur in the remaining blocks.

Proceeding in this way, we find the block list

$B_1 = (3, 7, 11, 15)$,
$B_2 = (2, 6, 10, 14)$,
$B_3 = (1, 5, 9, 13)$,
$B_4 = (0, 6, 10, 12)$,
$B_5 = (0, 7, 8, 15)$,
$B_6 = (0, 6, 8, 14)$,
$B_7 = (0, 7, 9, 14)$,
$B_8 = (0, 5, 8, 13)$,
$B_9 = (0, 7, 10, 13)$,
$B_{10} = (0, 4, 8, 12)$,
$B_{11} = (0, 7, 11, 12)$.

Fig. 5.17

Since this list satisfies the condition of Theorem 4.4, it generates a snake. Actually, the applied rules constitute a generalization of the well-ordering criterion as formulated in Corollary 4.3. We also remark that after having chosen B_6, the words with supports a until h do not play a role anymore. So, we only have to take care that when choosing B_i, the implications $i_2 = (i-1)_2 \Rightarrow i_4 \neq l_4$ and $i_4 = (i-1)_4 \Rightarrow i_2 \neq l_2$, $l > i$, are satisfied.

Finally, we present two more examples of block lists which generate a snake in Q_{16} of length 2^{13}. In the next chapter we shall apply these examples for the construction of a cover of Q_{16} with eight snakes.

Example 5.7

Consider the following list $\mathcal{B}$ of eleven independent blocks satisfying the fixed-position property and forming a basis for the Reed-Muller code $R(2, 4)$.

$B_1 = (\ 3, 5, 11, 13),$
$B_2 = (\ 2, 5, 11, 12),$
$B_3 = (\ 1, 5, 11, 15)$
$B_4 = (\ 0, 5, 11, 14),$
$B_5 = (\ 0, 6,\ \ 8, 14),$
$B_6 = (\ 0, 6, 10, 12),$
$B_7 = (\ 0, 7, 10, 13),$
$B_8 = (\ 0, 4, 11, 15).$
$B_9 = (\ 0, 7,\ \ 8, 15),$
$B_{10} = (\ 0, 6,\ \ 9, 15),$
$B_{11} = (\ 0, 6, 11, 13).$

Fig. 5.18

Again, the sublist $(B_1, B_2, \ldots, B_{10})$ is well-ordered. Adding B_{11} destroys that property. The only reason for this is that the integer $7_4 := 13$ now occurs in a block B_l with $l = 11 > 7$. Nevertheless, the list $\mathcal{B}$ defines a transition sequence $\overline{S}_{11}(\mathcal{B})$ which generates a snake of length 2^{13} in Q_{16}. To see this, one only has to verify that there are no codewords

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

of type (*i*) or (*ii*) (cf. Theorem 4.4 and Example 5.5) for $i = 8$. However, the only codewords of type (*i*) or (*ii*) which are generated by the basis vectors $\mathbf{b}_7$, $\mathbf{b}_8$, $\mathbf{b}_9$, $\mathbf{b}_{10}$ and $\mathbf{b}_{11}$ are these basis vectors themselves, as one can verify immediately. Since $\mathbf{b}_7$ is not of type (*i*) or (*ii*) ($7_2 \neq 8_2$ and $7_4 \neq 8_4$), we are done. That the above list really generates a snake in Q_{16} was confirmed by a computer program.

By similar arguments, we can show that the following list also generates a snake of length 2^{13} in Q_{16}.

$B_1 = (\ 3, 7, 11, 15),$
$B_2 = (\ 2, 6, 10, 14),$
$B_3 = (\ 1, 5,\ \ 9, 13),$
$B_4 = (\ 0, 4,\ \ 8, 12),$
$B_5 = (\ 0, 5,\ \ 8, 13),$

$B_6 = (\ 0, 6,\ \ 8, 14)$,
$B_7 = (\ 0, 7, 10, 13)$,
$B_8 = (\ 0, 7,\ \ 8, 15)$,
$B_9 = (\ 0, 7,\ \ 9, 14)$,
$B_{10} = (\ 0, 7, 11, 12)$.
$B_{11} = (\ 0, 6, 10, 12)$.

Fig. 5.19

5.6 A Sufficient Condition for a Block List to Generate a Snake

Inspired by Examples 5.5, 5.6 and 5.7, we are able to formulate a sufficient condition for a block list corresponding to an ordered basis of an $[n, k, 4]$-code $\mathcal{C}$, to construct a snake based on that block list. This condition is formulated only in terms of the blocks themselves, contrary to the conditions of Theorem 4.2 and Theorem 4.4.

Theorem 5.8

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a block list corresponding to an ordered minimum-weight basis of an $[n, k, 4]$-code $\mathcal{C}$, satisfying the fixed-position property. Let furthermore $\mathcal{B}$ be ordered lexicographically with respect to the integers of I_1, and let m be the integer as determined by Definition 4.2, $1 \leq m \leq k$. Then $\mathcal{B}$ can be transformed to a list $\mathcal{B}'$ corresponding to an equivalent basis of $\mathcal{C}$ such that $\mathcal{B}'$ generates a snake of length 2^{k+2}, if each block B_j of $\mathcal{B}$ with $m < j < k$, satisfies at least one of the following conditions :

(*i*) *the integer j_2 does not occur in any of the blocks B_l, $l > j$;*

(*ii*) *the integer j_4 does not occur in any of the blocks B_l, $l > j$;*

(*iii*) *the integer j_3 does not occur in any of the blocks B_l, $l > j$ and $|B_j \cap B_{j+1}| = 1$.*

Proof. We shall prove that the conditions of Theorem 4.4 can always be met by transforming $\mathcal{B}$ into an appropriate equivalent block list $\mathcal{B}'$.

From Definition 4.2(*i*) we know that the $m - 1$ integers $1_1, 2_1, \ldots, (m-1)_1$ are all distinct and that $m_1 = (m+1)_1 = \ldots = k_1$. If all blocks B_j, $m < j < k$, satisfy at least one of the conditions (*i*) and (*ii*), the list is well-ordered and does not need to be changed (cf. Corollary 4.3).

Suppose, there is a block B_i, $m < i$, which only satisfies condition (*iii*) and let there be a codeword (cf. (4.13))

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

with sup $\mathbf{c} = \{m_1, i_2, p_3, j_4\}$, $m < i < j < k$ (a support of type (*i*) in Theorem 4.4). This cannot happen if $(i-1)_2$ or $(i-1)_4$ does not occur in blocks B_l, $l \geq i$ (cf. the proof of Corollary 4.3). So, $(i-1)_3$ does not occur in blocks B_l, $l \geq i$, while $(i-1)_2$ and $(i-1)_4$ do. It follows that $p_3 = (i-1)_3$ and $p = i - 1$.

Now $\mathbf{c} \neq \mathbf{b}_{i-1}$, since $\mathbf{c} = \mathbf{b}_{i-1}$ would imply $(i-1)_2 = i_2$ contradicting $|B_j \cap B_{j+1}| = 1$. We define $\mathbf{b}'_{i-1} := \mathbf{c}$ and replace B_{i-1} in the block list $\mathcal{B}$ by B'_{i-1}, thus providing us with a new list $\mathcal{B}'$. This list $\mathcal{B}'$ also satisfies the condition that each block contains at least one integer that does not occur in blocks with higher index because $(i-1)_3 \in B'_{i-1}$ and $(i-1)_3 \notin B'_l$, $l \geq i$.

The relation $\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$ must now be written as $\mathbf{c}' := \mathbf{b}_{i-1} = \mathbf{b}'_{i-1} + \sum_{l \geq i} \mathbf{b}_l$ with $\sup \mathbf{c}' = \{m_1, (i-1)_2, (i-1)_3, (i-1)_4\}$. Since $i_2 \neq (i-1)_2$ and $i_4 \neq (i-1)_4$, this support is not of type (*i*) or (*ii*) with respect to the new list $\mathcal{B}'$.

We would have obtained a similar result if we had made the assumption

$$\sup \mathbf{c}' = \{m_1, j_2, i_3, i_4\},\ m < i < j < k.$$

Hence, by carrying out this process for $i = m+1, m+2, \ldots, k$, and if necessary replacing B_{i-1} by an equivalent block B'_{i-1}, we can transform $\mathcal{B}$ into a block list $\mathcal{B}'$ which satisfies the condition of Theorem 4.4. □

As is clear from the statement as well as from the proof , Theorem 5.8 is sufficient but not necessary for a block list to generate a snake. The list of Fig. 5.13 provides us wih a counter example.

Remark

The condition of Theorem 5.8 implies that the code $\mathcal{C}$ has a *minimum-weight basis in echelon form* with the additional property that if $j_3 \in I_3$ is the only pivot of $\mathbf{b}_j$, one has

$$\left\| \mathbf{b}_j + \mathbf{b}_{j+1} \right\| = 6.$$

For examples, we refer to Section 8.4.

6. Translations of Snakes

In this Chapter, we shall discuss a number of theorems dealing with so-called translations of snakes, where first a snake is constructed by the method of Chapter 5, and next this snake is translated over some fixed vector in $GF(2)^n$. More precisely, we shall derive criteria when such translated snakes are disjoint with each other. In the proofs, we shall exploit the (partial) linear character of the snakes produced by our method.

6.1 Conditions for Disjoint Translated Snakes

It is obvious that if we apply a transition sequence of a snake $\mathcal{S}$ starting with a word $\mathbf{t}$ different from the zeroword $\mathbf{0}$, we again obtain a snake. Since this snake can be obtained by addition of $\mathbf{t}$ to *all* words of $\mathcal{S}$, we denote it by $\mathcal{S} + \mathbf{t}$, and we call this snake a *translation* of $\mathcal{S}$ over the vector $\mathbf{t}$. More precisely, if we start with a snake

$$\mathcal{S} = (\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_L), \tag{6.1}$$

then

$$\mathcal{S} + \mathbf{t} = (\mathbf{w}_0 + \mathbf{t}, \mathbf{w}_1 + \mathbf{t}, \ldots, \mathbf{w}_L + \mathbf{t}). \tag{6.2}$$

An obvious question is whether $\mathcal{S}$ and $\mathcal{S} + \mathbf{t}$ are *disjoint* (i.e. whether they have no common words). In this section, we shall study this question with respect to the snakes which are produced by the method of Chapter 5, and which are based on a linear $[n, k, 4]$-code. In the next, the vector $\mathbf{e}_i$ stands for the unit vector with a 'one' on position i and zeros elsewhere, for $0 \le i \le n-1$, just like in Chapter 4 and 5, and we label the positions in a snakeword of length n from 0 until $n-1$, from left to right.

Throughout this Chapter, and also in Chapters 7 and 8, we shall use in the formulation of our theorems and in the text, labels $p_i, q_j, \ldots$, etc, for labeling unit vectors like $\mathbf{e}_{p_i}$, $\mathbf{e}_{q_j}$, …, etc. The meaning of this notation is that $p_i \in I_i$, $1 \le i \le 4$, is the i-th element in block $B_p = (p_1, p_2, p_3, p_4)$ and $q_j \in I_j$, $1 \le j \le 4$, is the j-th element in block $B_q = (q_1, q_2, q_3, q_4)$, for $p, q \in \{1, 2, \ldots, k\}$.

Theorem 6.1

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a block list corresponding to an ordered minimum-weight basis of an $[n,k,4]$-code $\mathcal{C}$ satisfying the fixed-position property. Let $\mathcal{S}$ be the snake defined by the transition sequence $\overline{S}_k(\mathcal{B})$. Then the snakes

$$\mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_3}$$

and

$$\mathcal{S} + \mathbf{e}_{p_2} + \mathbf{e}_{q_4}$$

are disjoint with $\mathcal{S}$ for all $p, q \in \{1, 2, \ldots, k\}$.

Proof. In order to prove that $\mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_3} \cap \mathcal{S} = \emptyset$, we shall prove the equivalent statement that there are no two vectors $\mathbf{x}$ and $\mathbf{y}$ in $\mathcal{S}$ such that $\mathbf{x} + \mathbf{y} = \mathbf{e}_{p_1} + \mathbf{e}_{q_3}$.

Assume that there exist such vectors $\mathbf{x}$ and $\mathbf{y}$. Writing $\mathbf{x} = \mathbf{c}' + \mathbf{z}'$, with $\mathbf{z}' \in \{\mathbf{0}, \mathbf{e}_{i_1}, \mathbf{e}_{i_1} + \mathbf{e}_{i_2}, \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3}\}$ and $\mathbf{y} = \mathbf{c}'' + \mathbf{z}''$, with $\mathbf{z}'' \in \{\mathbf{0}, \mathbf{e}_{j_4}, \mathbf{e}_{j_3} + \mathbf{e}_{j_4}, \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}\}$, and $\mathbf{c}', \mathbf{c}'' \in \mathcal{C}$, like in the proof of Theorem 4.2, and making use of the minimum weight 4 of $\mathcal{C}$, we obtain the following possibilities for $W := \sup \mathbf{c}$ with $\mathbf{c} = \mathbf{c}' + \mathbf{c}''$:

1. $W = \{i_1, p_1, q_3, j_4\}$;
2. $W = \{i_1, i_2, p_1, q_3\}$;
3. $W = \{p_1, q_3, j_3, j_4\}$;
4. $W = \{i_1, p_1, q_3, j_2, j_3, j_4\}$;
5. $W = \{i_1, i_2, i_3, p_1, q_3, j_4\}$;
6. $W = \{i_1, i_2, p_1, q_3, j_3, j_4\}$;
7. $W = \{i_1, i_2, i_3, p_1, q_3, j_2, j_3, j_4\}$,

where $i,j \in \{1, 2, \ldots, k\}$. However, since $\overline{S}_k(\mathcal{B})$ is constructed from a basis $\mathcal{B}$, (of $\mathcal{C}$) which satisfies the fixed position property, all these possibilities are eliminated by Lemma 3.1. So we may conclude that $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_3}$ are disjoint snakes for all $p, q \in \{1, 2, \ldots, k\}$. In the same way we can prove that $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_2} + \mathbf{e}_{q_4}$ are disjoint for all $p, q \in \{1, 2, \ldots, k\}$. □

We remark that for $p = q$, we obtain Theorem 5 of [34].

Instead of translations over vectors $\mathbf{e}_{p_1} + \mathbf{e}_{q_3}$ or $\mathbf{e}_{p_2} + \mathbf{e}_{q_4}$, we can study, more generally, translations over a vector $\mathbf{e}_{p_b} + \mathbf{e}_{q_c}$ with $b, c \in \{1, 2, 3, 4\}$, $p, q \in \{1, 2, \ldots, k\}$. First we remark that for $q > 1$, the transition sequence $\overline{S}_k(\mathcal{B})$ contains subsequences B_1B_q and B_qB_1 due to the definition of the standard Gray code. Hence, $\overline{S}_k(\mathcal{B})$ contains consecutive pairs of integers $1_4, q_1$ and q_4, i_1 and therefore $\mathcal{S}$ contains words (cf. the proof of Theorem 6.1) $\mathbf{x} = \mathbf{c} + \mathbf{e}_{1_4}$, $\mathbf{y} = \mathbf{c} + \mathbf{e}_{q_1}$ and $\mathbf{x} = \mathbf{c} + \mathbf{e}_{q_4}$, $\mathbf{y} = \mathbf{c} + \mathbf{e}_{1_1}$ (so $\mathbf{w} = \mathbf{c} + \mathbf{c} = \mathbf{0}$). This implies that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{1_4} + \mathbf{e}_{q_1} \neq \emptyset$ and also $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{q_4} \neq \emptyset$.

Theorem 6.2

Let $\mathcal{S}$ be the snake defined by $\overline{S}_k(\mathcal{B})$ as defined in Theorem 6.1. *Then the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$, $p_b \neq q_c$, are disjoint if the code $\mathcal{C}$ (generated by $\mathcal{B}$) does not contain any word $\mathbf{c}$ with support $W = \sup \mathbf{c}$ which is of one of the following types*:

(*i*). $\{i_1, p_b, q_c, j_4\}$;
(*ii*). $\{i_1, i_2, p_b, q_c\}$;
(*iii*). $\{i_1, p_b, q_c, j_1\}$;
(*iv*). $\{i_4, p_b, q_c, j_4\}$;
(*v*). $\{i_1, i_2, p_b, q_c, j_3, j_4\}$,

for $1 \leq i, j \leq k$.

Proof. We first remark that the five sets in the Theorem should be interpreted as *multisets* which may contain some elements more than once (cf. also Remark 6.1). It will be obvious that the relation $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c} = \varnothing$ is equivalent to the statement that there are no two vectors $\mathbf{x}$ and $\mathbf{y}$ in $\mathcal{S}$ such that $\mathbf{x} + \mathbf{y} = \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$. Assume that the snakes are not disjoint.

Writing $\mathbf{x} = \mathbf{c}' + \mathbf{z}'$ and $\mathbf{y} = \mathbf{c}'' + \mathbf{z}''$ and using the notations and arguments applied in the proof of Theorem 6.1, we obtain that $W := \sup \mathbf{c} = \sup \mathbf{c}' + \mathbf{c}''$ is one of the seven possibilities 1 until 7 in that proof with p_1 and q_3 replaced by p_b and q_c.

The possibilities 1, 2 and 6 are equivalent to (*i*), (*ii*) and (*v*), respectively. If $W = \{p_b, q_c, j_3, j_4\}$, then $\sup \mathbf{c} + \mathbf{b}_j = \{j_1, j_2, p_b, q_c\}$, which is already covered by (*ii*) since both i and j are arbitrary integers of $\{1, 2, \ldots, k\}$. If $W = \{i_1, p_1, q_3, j_2, j_3, j_4\}$ (Case 4), then $\sup \mathbf{c} + \mathbf{b}_j = \{i_1, p_b, q_c, j_1\}$, which is equivalent to (*iii*). Similarly, Case 5 is equivalent to Case (*iv*). Finally, if $\sup \mathbf{c} = \{i_1, i_2, i_3, p_b, q_c, j_2, j_3, j_4\}$, then $\sup \mathbf{c} + \mathbf{b}_i + \mathbf{b}_j = \{j_1, p_b, q_c, j_4\}$, and hence, Case 7 is equivalent to (*i*). □

Remark 6.1

Again, we can conclude that the pairs $\{1_1, q_4\}$ and $\{1_4, q_1\}$ will never give rise to disjoint snakes, since Theorem 2.8 (*ii*) guarantees the occurrence of a support $W = \{1_1, 1_1, q_4, q_4\}$ and $W = \{q_1, q_1, 1_4, 1_4\}$ which both correspond to $\mathbf{c} = \mathbf{0} \in \mathcal{C}$.

In the following example, we take special values for the indices b and c in Theorem 6.2.

Example 6.1

Take $b = 1$ and $c = 1$. A sufficient condition that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_1} = \varnothing$ is that $\mathcal{C}$ does not contain words of the following three types (among the five types stated in Theorem 6.2):

(iii) $\{i_1, p_1, q_1, j_1\}$;

(iv) $\{i_4, p_1, q_1, j_4\}$;

(v) $\{i_1, i_2, p_1, q_1, j_3, j_4\}$.

Types (*i*) and (*ii*) stated in Theorem 6.2 can be omitted in this case, because of Lemma 4.1.

Consider the snake $\mathcal{S}$ in Q_{16} defined by $\overline{S}_{11}(\mathcal{B})$, with $\mathcal{B}$ from Fig. 5.19. By computer search we found that $\mathbf{w}_{30}$ of $\mathcal{S}$ and $\mathbf{w}_{6462}$ of $\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_1$ are the same. By using the conversion rule which transforms i to $\mathbf{w}_i$, developed in Section 4.3, we shall compute these two words.

Since $30 = 4 \cdot 7 + 2$, we first compute

$$\mathbf{g}_7 = 0111 + 0011 = 0100$$

and so

$$\mathbf{w}_{28} = \mathbf{b}_3,$$

or

$$\sup \mathbf{w}_{28} = \{1, 5, 9, 13\}.$$

Similarly, we have

$$\mathbf{g}_8 = 1000 + 0100 = 1100,$$

and hence

$$\mathbf{w}_{32} = \mathbf{b}_3 + \mathbf{b}_4$$

or

$$\sup \mathbf{w}_{32} = \{0, 1, 4, 5, 8, 9, 12, 13\}.$$

It also follows that

$$\mathbf{w}_{30} = \mathbf{w}_{28} + \mathbf{e}_{4_1} + \mathbf{e}_{4_2}$$

or

$$\sup \mathbf{w}_{30} = \{0, 1, 4, 5, 9, 13\}.$$

Next, we compute $\mathbf{w}_{6462}$. Since $6462 = 4 \cdot 1615 + 2$, we find`

$$\mathbf{g}_{1615} = 11001001111 + 01100100111 = 10101101000$$

and so

$$\mathbf{w}_{6460} = \mathbf{b}_4 + \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_9 + \mathbf{b}_{11},$$

or

$$\sup \mathbf{w}_{6460} = \{0, 4, 9, 13\}.$$

When adding 1 to the binary representation of 1615, the carry stops at position 5. Therefore,

$$\mathbf{w}_{6462} = \mathbf{w}_{6460} + \mathbf{e}_{5_1} + \mathbf{e}_{5_2} = \mathbf{w}_{6460} + \mathbf{e}_0 + \mathbf{e}_1,$$

and hence

$$\sup \mathbf{w}_{6462} = \{4, 5, 9, 13\}.$$

So, we find indeed that

$$\mathbf{w}_{30} = \mathbf{e}_0 + \mathbf{e}_1 + \mathbf{w}_{6462}.$$

In this example, we have to write

$$\mathbf{x} = \mathbf{w}_{28} + \mathbf{e}_{4_1} + \mathbf{e}_{4_2},$$

and

$$\mathbf{y} = \mathbf{w}_{6464} + \mathbf{e}_{5_3} + \mathbf{e}_{5_4},$$

$$\mathbf{c} = \mathbf{w}_{28} + \mathbf{w}_{6464} = \mathbf{e}_{4_1} + \mathbf{e}_{4_2} + \mathbf{e}_0 + \mathbf{e}_1 + \mathbf{e}_{5_3} + \mathbf{e}_{5_4}$$

$$= \mathbf{e}_0 + \mathbf{e}_4 + \mathbf{e}_0 + \mathbf{e}_1 + \mathbf{e}_8 + \mathbf{e}_{13}.$$

This shows that $\mathcal{C}$ contains a word of type (v), and hence the condition of Theorem 6.2 is not satisfied. Therefore we are not entitled to apply this theorem.

Remark 6.2

We can not say at this moment that the occurrence of this word $\mathbf{c}$ with a support of type (v) proves that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_1 \neq \emptyset$, only by applying Theorem 6.2, since we did not prove that the sufficient condition of Theorem 6.2 is necessary as well. Probably, it is not. For a sufficient and necessary condition, we refer to Theorem 6.3, Theorem 6.6 and Example 6.2.

Remark 6.3

Considering Example 6.1, we have to take into account that in a support of type (v), it may happen that $i_1 = p_1$ and that the corresponding word is of weight 4, instead of weight 6.

As we already emphasized in Remark 6.2, the condition of Theorem 6.2 is a sufficient condition for the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$ being disjoint. Actually, we can formulate a stronger condition which is necessary as well.

From the proof of Theorem 6.2, it will be obvious that not only $\mathcal{C}$ must not contain words $\mathbf{c}$ with $W := \sup \mathbf{c}$ of one of the types $(i) - (v)$, but also that $W = c(B_i, \mathcal{T}', B_j)$, where B_i, $\mathcal{T}'$, B_j is a subsequence of $\overline{S}_k(\mathcal{B})$ such that the indices i and j are the same integers as those occurring in W. This extra requirement is really restrictive, since it can happen that $\mathcal{C}$ contains a word $\mathbf{c}$ with $\sup \mathbf{c} = \{i_1, p_b, q_c, j_4\}$ (type (i)) but that there is no subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ with $c(\mathcal{T}) = \{i_1, p_b, q_c, j_4\}$, due to the special properties of the external order of the blocks B_l, $1 \leq l \leq k$, imposed by the Gray code (in our case, the standard Gray code).

We emphasize that a similar remark cannot be made concerning the condition in Theorem 4.2. That condition is sufficient and necessary as well, due to the requirement that $\mathbf{c}$ is of the form (6.3), which shows a dependency on i ($< j$).

Theorem 6.3

Let $\mathcal{S}$ be the snake defined by $\overline{S}_k(\mathcal{B})$ like in Theorem 6.1. *Then the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$, $p_b \neq q_c$, are disjoint if and only if $\overline{S}_k(\mathcal{B})$ contains no subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ with $c(\mathcal{T})$ is of one of the following types*

(i) $\{i_1, p_b, q_c, j_4\}$;

(ii) $\{i_1, i_2, p_b, q_c\}$;

(iii) $\{i_1, p_b, q_c, j_2, j_3, j_4\}$;

(iv) $\{i_1, i_2, i_3, p_b, q_c, j_4\}$;

(v) $\{i_1, i_2, p_b, q_c, j_3, j_4\}$,

for $1 \leq i, j \leq k$.

Proof. The proof of the *if*-part is similar to the proof of Theorem 6.2. Assume that the condition of the Theorem holds. Suppose that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c} \neq \varnothing$. Then there exists a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ of $\overline{S}_k(\mathcal{B})$ such that $W := c(\mathcal{T})$ is equal to one of the possibilities 1 until 7 mentioned in the proof of Theorem 6.1 with p_1 and q_3 replaced by p_b and q_c, respectively. However, possibilities 1, 2, 4, 5 and 6 correspond to (i), (ii), (iii), (iv) and (v), which do not occur by the conditions of the Theorem.

Suppose that $W = \{p_b, q_c, j_3, j_4\}$ (possibility 3). If we consider $\overline{\mathcal{T}} = B_i, \mathcal{T}'$ (the sequence $\mathcal{T}$ without its last block), then $\overline{W} = c(\overline{\mathcal{T}}) = \{j_1, j_2, p_b, q_c\}$. Now the complement of $\overline{\mathcal{T}}$ in $\overline{S}_k(\mathcal{B})$ is a subsequence

$$\overline{\mathcal{T}^c} = B_j, \mathcal{T}'', B_l$$

with $c(\overline{\mathcal{T}^c}) = \overline{W} = \sup(\mathbf{c} + \mathbf{b}_j)$. However, such a subsequence $\overline{\mathcal{T}^c}$ of $\overline{S}_k(\mathcal{B})$ does not exist, by assumption, since it is of type (*ii*). In a similar way we can eliminate possibility 7. Let $\mathbf{c} \in \mathcal{C}$ have a support $W = \{i_1, i_2, i_3, p_b, q_c, j_2, j_3, j_4\} = c(\mathcal{T}) = c(B_i, \mathcal{T}', B_j)$. Then $\sup \mathbf{c} + \mathbf{b}_i + \mathbf{b}_j = \overline{W} = \{j_1, p_b, q_c, i_4\} = c(B_j, \mathcal{T}^c, B_i)$. This is already covered by (*i*), since both *i* and *j* can take on any value from $\{1, 2, \ldots, k\}$.

As for the *only-if* part, we have to show that if the two snakes have an empty intersection, there does not exist a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ with $c(\mathcal{T})$ of type (*i*), (*ii*), …, (*v*). Suppose there does exist a $\mathcal{T}$ with $c(\mathcal{T})$ of type (*i*). Due to the structure of $\mathcal{T}$, it has a subsequence $\overline{\mathcal{T}} = i_2, i_3, i_4, \mathcal{T}', j_1, j_2, j_3$ with $c(\overline{\mathcal{T}}) = \{i_1, p_b, q_c, j_4\} \backslash \{i_1, j_4\} = \{p_b, q_c\}$. So, there exist words $\mathbf{x}$ and $\mathbf{y}$ in $\mathcal{S}$ such that $\mathbf{x} + \mathbf{y} = \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$, and hence the snakes are not disjoint, which contradicts our assumption. If $\mathcal{T}$ is of one of the other types (*ii*) – (*v*), we can argue similarly. We conclude that the condition of the Theorem is also necessary. □

Like in Theorem 6.2, we can have that in cases (*iii*), (*iv*) and (*v*), one of the indices *i*- or *j*-indices is equal to p_b or q_c. If this happens then the corresponding codeword $\mathbf{c}$ has weight 4 instead of 6. One can interpret Theorem 6.2 as a corollary of Theorem 6.3.

The condition of Theorem 6.3 is formulated in terms of subsequences of $\overline{S}_k(\mathcal{B})$. A formulation in terms of codewords of $\mathcal{C}$, like in Theorem 4.2, seems unattractive in the *general case*, because of the various possibilities for *i*, *j*, p_b and q_c. Instead, we shall focus on special choices for the pair $\{p_b, q_c\}$.

In the next, $\mathcal{S}$ will always be a snake defined by the transition sequence $\overline{S}_k(\mathcal{B})$ of Theorem 4.2, while *p* and *q* are integers from $\{1, 2, \ldots, k\}$. Furthermore, $T := i, T', j$ will be the subsequence of the transition sequence of the standard Gray code $G(k)$, the elements of which represent the indices of the blocks $B_i, \ldots, B_j$ in $\mathcal{T} = B_i, \mathcal{T}', B_j$, and $A := c(\mathcal{T})$.

Special cases are the following two corollaries.

Corollary 6.4

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_3} = \varnothing \qquad \textit{and} \qquad \mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_2} + \mathbf{e}_{q_4} = \varnothing.$$

Proof. For $b = 1$ and $c = 3$, none of the types (*i*) – (*v*) in Theorem 6.3 satisfies Lemma 4.1. The same is true for $b = 2$ and $c = 4$. □

Corollary 6.5

If $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *is well-ordered and if* $p_3 \neq q_3$, *then* $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} = \varnothing$.

Proof. Suppose that the above intersection is not empty. For $b = 3$ and $c = 3$, only types (*iii*), (*iv*) and (*v*) of Theorem 6.3 can occur, because of Lemma 4.1. Assume that $\mathbf{c} \in \mathcal{C}$ has support W of type (*iv*). It will be obvious, due to the fixed-position property of the basis vectors of $\mathcal{C}$, that in this case $\mathbf{c}$ is the sum of an odd number of basis vectors. So, $|A|$ (cf. eq. (10)) is odd and it follows that $i \neq j$, $1 < i$, $1 < j$ (Theorem 2.5(*iv*)). Just as in Chapter 4, we assume that $i < j$. Then we have according to the same Theorem that $i - 1 = min\ A$, and we can write (cf. (4.13))

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l \ , \tag{6.3}$$

where l runs through the set $A \setminus \{i - 1\}$. Since the blocks in $\mathcal{B}$ are well-ordered, at least one of the integers $(i - 1)_1$, $(i - 1)_2$ and $(i - 1)_4$ does not occur in blocks B_l, $l \geq i$.

If $(i - 1)_1$ does not occur anymore, then we have from (6.3) that $(i - 1)_1 \in \text{sup}\ \mathbf{c}$, and so $(i - 1)_1 = i_1$, which is a contradiction with the assumption that $(i - 1)_1$ does not occur anymore. A similar contradiction follows if $(i - 1)_2$ or $(i - 1)_4$ does not occur anymore. In the case $1 < j < i$, we can derive similar contradictions by considering $\mathbf{c} + \mathbf{b}_i + \mathbf{b}_j$, the support $\{j_1, j_2, j_3, p_3, q_3, i_4\}$ of which corresponds to the subsequence $B_j, \mathcal{T}^c, B_i$ of $\overline{S}_k(\mathcal{B})$. We conclude that type (*iv*) does not occur. Similarly, we can prove that types (*iii*) and (*v*) can not occur. Here, we emphasize that a word $\mathbf{c} \in \mathcal{C}$ with support of type (*iii*) or (*v*) is again the sum of an odd number of basis vectors. □

6.2 Special Cases

In this subsection, we study the intersection

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_b} + \mathbf{e}_{q_c}$$

for some special choices of p_b and q_c.

Theorem 6.6

Let $\mathcal{S}$ be the snake defined by $\overline{S}_k(\mathcal{B})$ as in Theorem 6.1. *Then the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_1}$, $p_1 \neq q_1$, are disjoint if and only if there is no codeword $\mathbf{c} \in \mathcal{C}$ as expressed by* (6.3), *such that its support W is one of the sets*

(*i*). $\{i_1, p_1, q_1, j_1\}$;

(*ii*) $\{i_4, p_1, q_1, j_4\}$;

(*iii*) $\{i_1, i_2, p_1, q_1, j_3, j_4\}$,

for some j, with $1 < i < j \leq k$.

Proof. Assume that the condition of the Theorem holds. If $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_1} \neq \emptyset$, then there is a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ of $\overline{S}_k(\mathcal{B})$ such that $c(\mathcal{T})$ is of one of the types (*i*) – (*v*) in Theorem 6.3. Now, types (*i*) and (*ii*) cannot occur, because of Lemma 4.1. Suppose that $c(\mathcal{T})$

$= \{i_1, i_2, p_1, q_1, j_3, j_4\}$ (type (v)). Since $\mathcal{T}$ consists of complete blocks, $c(\mathcal{T}) = W = \sup \mathbf{c}$, for some $\mathbf{c} \in \mathcal{C}$. It follows that $i \neq j$, because otherwise the weight of $\mathbf{c} + \mathbf{b}_i$ is 2 and $\mathcal{C}$ does not contain words of weight 2. It is also obvious that $\mathbf{c}$ is the sum of an odd number of basis vectors, due to the fixed-position property satisfied by $\mathcal{B}$. So, $1 < i$ and $1 < j$ (cf. Theorem 2.5).

We write

$$\mathbf{c} = \sum_{l \in A} \mathbf{b}_l ,$$

with $|A|$ is odd. For $i < j$ we have that $i - 1 = \min A$ (Theorem 2.5 (i)), and therefore we can write

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l ,$$

where l runs through the set $\overline{A} = A \setminus \{i - 1\}$, which is of even size. This expression is identical to (6.3), and so we have a contradiction. For $j < i$, we consider the subsequence $B_j, \mathcal{T}^c, B_i$, where $\mathcal{T}^c$ is the complement of $\mathcal{T}$ in $\overline{S}_k(\mathcal{B})$ which has the same contents as $\mathcal{T}$ itself.

We now consider $\mathbf{c}' = \mathbf{c} + \mathbf{b}_i + \mathbf{b}_j$ which has support $\sup \mathbf{c}' = c(B_j, \mathcal{T}^c, B_i) = \{j_1, j_2, p_1, q_1, i_3, i_4\}$. By interchanging i and j, we obtain a case which is identical to the previous one. So, again we have a contradiction with the condition of the Theorem.

If $c(\mathcal{T}) = \{i_1, p_1, q_1, j_2, j_3, j_4\}$ (type (iii) of Theorem 6.3), it follows again that $i \neq j$, $1 < i$, $1 < j$. If $1 < i < j$, then there exists a $\mathbf{c} \in \mathcal{C}$ of the form (6.3), with $\sup \mathbf{c} = c(\mathcal{T})$ and

$$\mathbf{c} = \sum_{l \in A} \mathbf{b}_l = \mathbf{b}_{i-1} + \sum_{l \in \overline{A}} \mathbf{b}_l ,$$

where $A = \sup \mathbf{c} = c(\mathcal{T})$, $\overline{A} = A \setminus \{i - 1\}$, with $|A|$ is odd. If we define $\mathbf{c}' = \mathbf{c} + \mathbf{b}_j$, we have equivalently that $\sup \mathbf{c}' = \{i_1, p_1, q_1, j_1\}$,

$$\mathbf{c}' = \mathbf{b}_{i-1} + \sum_{l \in \overline{A'}} \mathbf{b}_l ,$$

where $\overline{A'} = A \cup \{j\} \setminus \{i - 1\}$, for $j \notin A$, and $\overline{A'} = A \setminus \{i - 1, j\}$, for $j \in A$. So, we find again a contradiction with the condition of the Theorem. In the case $1 < j < i$, we can argue similarly, by interchanging i and j. If $c(\mathcal{T})$ is of type (iv) of Theorem 6.3, we can obtain a contradiction in completely the same way. So, the *if* part is proved.

To prove the *only-if* part, we can apply Theorem 2.7 (i) and (ii), to show that if there exists a codeword $\mathbf{c} \in \mathcal{C}$ as described by the Theorem, it follows that there exists a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ such that $\sup \mathbf{c} = c(\mathcal{T})$ (cf. the proof of Theorem 4.2). By reversing the arguments used in the *if* part of the proof and applying Theorem 6.3, it then follows that the two snakes are not disjoint. □

Example 6.2.

In Example 6.1, we found that if $\mathcal{S}$ is defined by the ordered block list of Fig. 5.19, we have that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_1 \neq \emptyset$, or equivalently $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{3_1} + \mathbf{e}_{4_1} \neq \emptyset$. This result now follows

from Theorem 6.6. If we define $\mathbf{c} = \mathbf{b}_3 + \mathbf{b}_4 + \mathbf{b}_5 + \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_9 + \mathbf{b}_{11}$ then

$$\sup \mathbf{c} = \{1, 4, 8, 13\}.$$

If we interpret this set as the multiset $\{0, 4, 1, 0, 8, 13\} = \{4_1, 4_2, 3_1, 4_1, 5_3, 5_4\}$, we see that sup $\mathbf{c}$ is of type $\{i_1, i_2, p_1, q_1, j_3, j_4\}$, and so the condition of the Theorem is not satisfied. Notice that $i - 1 = \min A = \min \{3, 4, 5, 6, 7, 9, 11\} = 3$, and hence $i = 4 < 5 = j$.

There is also a codeword $\mathbf{c}' \in \mathcal{C}$ such that sup $\mathbf{c}'$ is of type $\{i_1, p_1, q_1, j_1\}$. If we define $\mathbf{c}' = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3 + \mathbf{b}_5 + \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11}$, then sup $\mathbf{c}' = \{2, 1, 0, 3\} = \{2_1, 3_1, 4_1, 1_1\}$, and $i - 1 = \min A = 1$, $i = 2$. However, $j = 1$ and so $i > j$ which implies that we are not allowed to apply Theorem 6.6..

Example 6.3.

Let $\mathcal{S}$ be the snake defined by the ordered block list of Fig. 5.19. Now, we take $p_1 = 1_1$, $q_1 = 2_1$ and we consider the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{2_1}$. If we write the support of $\mathbf{c}'$ from the previous Example, as sup $\mathbf{c}' = \{1, 3, 2, 0\} = \{3_1, 1_1, 2_1, 4_1\}$ and interpret this set as $\{i_1, p_1, q_1, j_1\}$, it will be obvious that the word $\mathbf{c}'$ cannot be written in the form (6.3), since $i = 3$, and hence $i - 1 = 2 \neq \min A = 1$.

The last inequality holds because the block list in Fig. 5.19, though not well ordered, has the property $p_1 = 1_1$ ($\equiv 3$) only occurs in B_1. So, 1 is an element of A, where $\mathbf{c}' = \sum_{l \in A} \mathbf{b}_l$. It follows that there is no word in $\mathcal{C}$ of the first type mentioned in Theorem 6.6. However, there do exist words of type (*iii*) in Theorem 6.6. If we define $\mathbf{c} = \mathbf{b}_1 + \mathbf{b}_7 + \mathbf{b}_8 + \mathbf{b}_{10} + \mathbf{b}_{11}$, we find that sup $\mathbf{c} = \{3, 6, 8, 13\} = \{1_1, 2_2, 5_3, 5_4\} = \{2_1, 2_2, 1_1, 2_1, 5_3, 5_4\}$, and hence $i = 2$ and $j = 5$. Therefore,

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{2_1} \neq \varnothing.$$

Computer results show that the 511-rd word of $\mathcal{S}$ and the 4615-th word of $\mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{2_1}$ are both equal to 0000000010100100.

Likewise, we can prove

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{3_1} \neq \varnothing,$$
$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{1_1} + \mathbf{e}_{4_1} \neq \varnothing$$
$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{2_1} + \mathbf{e}_{3_1} \neq \varnothing$$

and

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{2_1} + \mathbf{e}_{4_1} \neq \varnothing.$$

The last inequality follows from the existence of the codeword $\mathbf{c} = \mathbf{b}_2 + \mathbf{b}_3 + \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_9$ which has a support $\{1, 5, 2, 0, 8, 14\} = \{3_1, 3_2, 2_1, 2_4, 6_3, 6_4\}$ of type (*iii*) in Theorem 6.6. Indeed a computer calculation show that the word $\mathbf{w}_{110}$ and $\mathbf{w}_{1662}$ (of $\mathcal{S}$) differ by $\mathbf{e}_2 + \mathbf{e}_0$.

Theorem 6.7

Let $\mathbf{S}$ *be the snake defined by* $\overline{S}_k(\mathcal{B})$ *as in Theorem* 6.1. *Then the snakes* $\mathbf{S}$ *and* $\mathbf{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_4}$ *are disjoint if and only if there is no codeword* $\mathbf{c} \in \mathcal{C}$ *as expressed by* (6.3), *such that its support W is one of the sets*

(*i*). $\{1_1, p_1, q_4, j_4\}, \quad j > 1;$

(*ii*) $\{i_1, p_1, q_4, 1_4\}, \quad i > 1.$

Proof. We take $p_b = p_1$, $q_c = q_4$. Applying Lemma 4.1 to Theorem 6.3, the only possibility for $\mathbf{S}$ and $\mathbf{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_4}$ having a non-empty intersection is that there is a subsequence $\mathcal{T} = B_i, \mathcal{T}'$, B_j such that $c(\mathcal{T}) = \{i_1, p_1, q_4, i_4\}$. Let $\mathbf{c}$ be the codeword of $\mathcal{C}$ with sup $\mathbf{c} = c(\mathcal{T})$. It is obvious that $\mathbf{c}$ is the sum of an even number of basiswords of $\mathbf{B}$. So, either $i = 1$ or $j = 1$ (Theorem 2.7), and so either sup $\mathbf{c} = \{1_1, p_1, q_4, j_4\}, j > 1$, or sup $\mathbf{c} = \{i_1, p_1, q_4, 1_4\}, i > 1$. □

6.3 Disjoint Snakes Based on Lexicographically Ordered Lists

To obtain disjoint snakes $\mathbf{S}$ and $\mathbf{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3}$ and to construct covers of Q_n, we cannot always apply Corollary 6.5, since not all block lists are well-ordered. If instead of being well-ordered, we only require a block list to be lexicographically ordered, we obtain a statement similar to Theorem 3.6.

Theorem 6.8

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *be a block list corresponding to an ordered minimum-weight basis of an* $[n,k,4]$*-code* $\mathcal{C}$ *satisfying the fixed-position property. Let furthermore* $\mathcal{B}$ *be ordered lexicographically according to condition* (*i*) *of Definition* 4.2 *If* $\overline{S}_k(\mathcal{B})$ *generates a snake* $\mathbf{S}$ *and if* $p_3 \neq q_3$, *then*

$$\mathbf{S} \cap \mathbf{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} = \varnothing,$$

if and only if there is no codeword $\mathbf{c} \in \mathcal{C}$ *as expressed by* (6.3) *with sup* $\mathbf{c}$ *equal to one of the following sets*

(i) $W = \{p_3, q_3, i_4, j_4\}$;

(ii) $W = \{i_2, j_2, p_3, q_3\}$,

for $m < i < j \leq k$.

Proof. First we prove the necessity of the conditions. Suppose there is a $\mathbf{c} \in \mathcal{C}$, written as

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

with sup $\mathbf{c} = W = \{p_3, q_3, i_4, j_4\}$, and $m < i < j$. Then $\mathbf{c}' = \mathbf{c} + \mathbf{b}_i$ has support $W' = \{i_1, i_2, i_3, p_3, q_3, j_4\}$ and we can write

$$\mathbf{c}' = \mathbf{b}_{i-1} + \sum_{l \geq i}{}' \mathbf{b}_l$$

and since $|W \cap I_j|$ is odd for $j \in \{1, 2, 3, 4\}$, $\mathbf{c}'$ is the sum of an odd number of blocks. Let A be the subset of the index set of the basis vectors which participate in the above expression for $\mathbf{c}'$, then $\min A = i - 1$.

According to Theorem 2.7, there is a subsequence $T = i, T', j$ of $\overline{S}_k$ (the transition sequence of the standard Gray code of length k) such that $A = c(\mathcal{T})$. Hence, $\overline{S}_k(\mathcal{B})$ contains a corresponding subsequence $\mathcal{T}$ such that $c(\mathcal{T}) = \sup \mathbf{c}' = W'$. From Theorem 6.3 (*iv*), it now follows that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} \neq \emptyset$. A similar proof can be given if there is

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

with $\sup \mathbf{c} = W = \{i_2, j_2, p_3, q_3\}$ and $m < i < j$.

In order to prove the *if*-part, we assume that $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} \neq \emptyset$. According to Theorem 6.3, there exists a subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ of $\overline{S}_k(\mathcal{B})$ such that $W := c(\mathcal{T})$ is equal to one of the sets (*i*) – (*v*). For $p_b = p_3$, $q_c = q_3$ it is immediately clear, by Lemma 4.1, that cases (*i*) and (*ii*) of Theorem 6.3 are impossible. Assume that W is of the form (*iii*). This implies that there is a codeword $\mathbf{c} \in \mathcal{C}$ with $\sup \mathbf{c} = W = \{i_1, p_3, q_3, j_2, j_3, j_4\}$. Equivalently, there exists a codeword $\mathbf{c}' = \mathbf{c} + \mathbf{b}_j$ with $W' = \sup \mathbf{c}' = \{i_1, j_1, p_3, q_3\}$.

For $i < j$ we can write $\mathbf{c}' = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$, where the summation is over an odd number of l-values. If $i \leq m$ it follows that $(i-1)_1 \in W'$, which contradicts $(i-1)_1 \neq i_1$ and $(i-1)_1 \neq j_1$. If $m < i$, we would have $i_1 = j_1$ yielding $W' = \{p_3, q_3\}$, which contradicts the minimum-weight of $\mathcal{C}$. For $j < i$ we can derive a similar contradiction.

Assume that W is of the form (*iv*). This implies that there is a codeword $\mathbf{c} \in \mathcal{C}$ with $\sup \mathbf{c} = W = \{i_1, i_2, i_3 p_3, q_3, j_4\}$ wich implies $i \neq j$ and $\mathbf{c}$ is the sum of an odd number of basis vectors. Equivalently, there exists a codeword $\mathbf{c}' = \mathbf{c} + \mathbf{b}_i$ with $W' = \{p_3, q_3, i_4, j_4\}$. Just as in the previous case, we can show that $m \leq i$ and also $m \leq j$. Because of condition (*i*) of the theorem, such a codeword $\mathbf{c}'$ does not exist for $i < j$, since we can write

$$\mathbf{c}' = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l + \mathbf{b}_i = \mathbf{b}_{i-1} + \sum_{l \geq i}{}' \mathbf{b}_l$$

in that case. If $j < i$, we can interchange the role of i and j.

Finally, we assume that W is of the form (*v*) of Theorem 6.3, i.e. $W = \{i_1, i_2, p_3, q_3, j_3, j_4\}$, $i \neq j$, or equivalently, there exists a codeword $\mathbf{c}' = \mathbf{c} + \mathbf{b}_j$ with $W' = \{i_1, i_2, j_1, j_2, p_3, q_3,\}$. Similarly, as in the previous case, we derive that that $m < i$ and also $m < j$. But then we would have $i_1 = j_1$, and hence $W' = \{i_2, j_2, p_3, q_3\}$. From condition (*ii*) of the theorem it follows that this is false, for $i < j$ as well as for $j < i$, just as in the previous case. □

In a completely similar way, we can prove the following theorem.

Theorem 6.9

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ *be a block list satisfying the conditions of Theorem* 6.8. *If* $\overline{S}_k(\mathcal{B})$ *generates a snake* $\mathcal{S}$ *and if* $p_2 \neq q_2$, *then*

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_2} + \mathbf{e}_{q_2} = \emptyset$$

if and only if there is no codeword $\mathbf{c} \in \mathcal{C}$ *as expressed by* (6.3) *with* $W = sup\ \mathbf{c}$ *equal to one of the following sets*

$$(i)\ W = \{p_2, q_2, i_4, j_4\}$$

$$(ii)\ W = \{i_2, j_2, p_2, q_2\}$$

for $m < i < j \leq k$, *and for any* p *and* q *with* $m - 1 \leq p, q \leq k$.

Example 6.4

Consider the block list of Fig. 5.19 and take

$$\{p_2, q_2\} = \{5, 7\}.$$

The only possible sets W of type (i) in Theorem 6.9 are {5, 7, 12, 14} and {5, 7, 13, 15} (Here, we applied the rule that the Nim sum of the integers in W must be equal to 0, because $\mathbf{c} \in \mathcal{C}$, cf. Section 5.4). In the first case, we have $\mathbf{c} = \mathbf{b}_5 + \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_{11}$ which implies $i = 6$, $i_4 = 6_4 = 14$, and hence $j_4 = 12$ and $j = 10$ or 11. So, there really exists a codeword as described in Theorem 6.9 and therefore,

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_5 + \mathbf{e}_7 \neq \emptyset.$$

If $\{p_2, q_2\} = \{6, 7\}$, then the only possible sets W of type (i) in Theorem 6.9 are {6, 7, 12, 13} and {6, 7, 14, 15}. In the first case, we have $\mathbf{c} = \mathbf{b}_7 + \mathbf{b}_{11}$, so $i = 8$, $i_4 = 8_4 = 15 \notin W$, while in the second case $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_8$, $i = 7$, $i_4 = 7_4 = 13 \notin W$. The only possible set W of type (ii) is {6, 7, 4, 5} implying $i = 5$, $i_2 = 5_2 = 5$, $j_2 = 4$ and so $j = 4$, which contradicts the condition $j > i$. So there is no $\mathbf{c}$ as described in Theorem 6.9, and it follows that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_6 + \mathbf{e}_7 = \emptyset.$$

In a similar way, we found

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_4 + \mathbf{e}_5 \neq \emptyset,$$

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_4 + \mathbf{e}_6 \neq \emptyset,$$

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_4 + \mathbf{e}_7 \neq \emptyset,$$

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_5 + \mathbf{e}_6 \neq \emptyset.$$

Instead of translating a snake over a vector of weight 2, one can also consider translations over vectors with a different weight value. After a moment's reflection, it is clear that a snake $\mathcal{S} + \mathbf{t}$ with $wt\ \mathbf{t} = 1$ will never be disjoint with $\mathcal{S}$, since for $\mathbf{t} = \mathbf{e}_p$, $p \in \{0, 1, \ldots, 15\}$, there is always a codeword $\mathbf{c} \in \mathcal{C}$ such that $\mathbf{c} + \mathbf{e}_p \in \mathcal{S}$. The same holds for any $\mathbf{t}$ of odd weight. As for vectors $\mathbf{t}$ of weight 4, we know for sure that $\mathcal{S} \cap \mathcal{S} + \mathbf{t} \neq \emptyset$ if $\mathbf{t} \in \mathcal{C}$. This is because if $\mathbf{t} \in \mathcal{C}$, then $\mathbf{c} + \mathbf{t} \in \mathcal{C}$ for $\mathbf{c} \in \mathcal{C}$ and both $\mathbf{c}$ and $\mathbf{c} + \mathbf{t}$ are words of $\mathcal{S}$. Therefore we investigate translations over $\mathbf{t}$ with $wt\ \mathbf{t} = 4$ and $\mathbf{t} \notin \mathcal{C}$.

The following theorem concerns vectors $\mathbf{t}$ with sup $\mathbf{t} = \{p_1, q_2, r_3, s_4\}$, $p_1 \in I_1$, $q_2 \in I_2$, $r_3 \in I_3$, $s_4 \in I_4$ and $p_1 \oplus q_2 \oplus r_3 \oplus s_4 \neq 0$. This last condition guarantees that $\mathbf{t} \notin \mathcal{C}$.

Theorem 6.10

Let $\mathcal{S}$ be the snake defined by $\overline{S}_k(\mathcal{B})$ as defined in Theorem 4.2. *Then the snakes $\mathcal{S}$ and $\mathcal{S} + \mathbf{e}_{p_1} + \mathbf{e}_{q_2} + \mathbf{e}_{r_3} + \mathbf{e}_{s_4}$, $p_1 \oplus q_2 \oplus r_3 \oplus s_4 \neq 0$, are disjoint if and only if $\overline{S}_k(\mathcal{B})$ contains no subsequence $\mathcal{T} = B_i, \mathcal{T}', B_j$ where $c(\mathcal{T})$ is of one of the following types.*

$$(i)\ \{i_1, p_1, q_2, r_3, s_4, j_2, j_3, j_4\};$$
$$(ii)\ \{i_1, i_2, p_1, q_2, r_3, s_4, j_1, j_2\};$$
$$(iii)\ \{i_1, i_2, i_3, p_1, q_2, r_3, s_4, j_4\},$$

for $1 \leq i, j \leq k$.

The proof is similar to the proof of Theorem 6.3.

Example 6.5

With respect to the list of Fig. 5.19, there does exist a codeword $\mathbf{c} = \mathbf{b}_1 + \mathbf{b}_6 + \mathbf{b}_8 + \mathbf{b}_{10}$ with sup $\mathbf{c} = \{0, 3, 6, 7, 12, 14\} = \{3, 0, 7, 10, 12, 6, 10, 14\}$ which is of type (*iii*) with $i = 1$ and $j = 2$. So, we may conclude that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_7 + \mathbf{e}_{10} + \mathbf{e}_{12} \neq \varnothing.$$

Indeed, taking $\mathbf{c}' = \mathbf{w}_0 \in \mathcal{C}$ and $\mathbf{c}'' = \mathbf{w}_{3320} = \mathbf{b}_1 + \mathbf{b}_6 + \mathbf{b}_8 + \mathbf{b}_{10}$, $\mathbf{x} = \mathbf{c}' + \mathbf{e}_{1_1} = \mathbf{c}' + \mathbf{e}_3$, $\mathbf{y} = \mathbf{c}'' + \mathbf{e}_{2_2} + \mathbf{e}_{2_3} + \mathbf{e}_{2_4} = \mathbf{c}'' + \mathbf{e}_6 + \mathbf{e}_{10} + \mathbf{e}_{14}$ yields $\mathbf{x} + \mathbf{y}$ with sup $\mathbf{x} + \mathbf{y} = \{0, 7, 10, 12\}$. Further calculations show that for each vector $\mathbf{t}$ with support $\{p_1, q_2, r_3, s_4\}$, $p_1 \oplus q_2 \oplus r_3 \oplus s_4 \neq 0$, there is always a codeword $\mathbf{c} \in \mathcal{C}$ which has a support of type (*i*), (*ii*) or (*iii*). Therefore, we have for the snake $\mathcal{S}$ defined by $\mathcal{B}$ in Fig. 5.19,

$$\mathcal{S} \cap \mathcal{S} + \mathbf{t} \neq \varnothing,$$

for all $\mathbf{t}$ with sup $\mathbf{t} = \{p_1, q_2, r_3, s_4\}$.

7. Covers and Near-Covers of Q_n, for $2 \leq n \leq 16$

As we remarked already in the Introduction, one major goal in this thesis is to construct families of mutual disjoint snakes of equal length, which together contain all binary words of length n. Since these binary words are the vertices of the hypercube Q_n and since a snake can be interpreted as a simple circuit in Q_n which satisfies the separability condition (cf. Definition 2.2 and 2.3) we speak of a cover of Q_n by a family of disjoint snakes. If p is the number of snakes of a family covering Q_n, we speak of a *p-cover* of Q_n, and if the snakes are symmetric, i.e. if their transition sequences are symmetric, we call it a *symmetric p-cover*.

In Section 7.1 we shall construct symmetric 4-covers for small hypercubes Q_n, $4 \leq n \leq 8$ and symmetric 8-covers Q_n, $8 \leq n \leq 15$. The constructions are based on block lists which are well-ordered, and so we can apply Corollary 6.5.

In Section 7.2, we try a similar construction to obtain an 8-cover for Q_{16} starting from the block list in Fig. 5.19. This list is no longer well-ordered, but since it is still lexicographically ordered, we can apply the results of Section 6.3. It turns out that we cannot obtain a complete 8-cover of Q_{16}, when starting from the list in Fig. 5.19. We call the resulting structures *near-covers*, since only very few vertices of Q_{16} are not incident with the union of the snakes in the family.

In Section 7.3, it is shown that there does exist a symmetric 8-cover for Q_{16} by taking the block list of Fig. 5.18, instead of Fig. 5.19 which corresponds to a different basis of $R(2, 4)$. By their very construction, all these families of snakes covering or nearly covering Q_n are invariant under the action of a certain group of translations. For this reason we also speak of a set of *parallel* snakes (nearly) covering Q_n, which has this group as *invariance* or *symmetry group*.

7.1 Symmetric Covers of Q_n, for $2 \leq n \leq 15$

It will be obvious that the square Q_2 has a 1-cover since the graph Q_2 itself is a snake. It is also obvious that the cube Q_3 cannot be covered by only one snake, since the maximal snake in Q_3 is of length $s(3) = 6$. However, Q_3 can be covered by two snakes lying in two (vertex)-disjoint Q_2-subgraphs of Q_3. We say that such a cover is obtained by *doubling* the 1-cover of Q_2.

A 2-cover of Q_4 can be obtained from the snake $\mathcal{S}^{(4)}$ generated by the transition sequence 0, 1, 2, 3, 0, 1, 2, 3. This snake can be obtained from the snake with transition sequence $\overline{S}_1(B_2')$ at the end of Section 5.4, by relabeling the bit positions. We translate this snake over the vector $\mathbf{e}_0 + \mathbf{e}_2$. According to Corollary 6.4, the snakes $\mathcal{S}^{(4)}$ and $\mathcal{S}^{(4)} + \mathbf{e}_0 + \mathbf{e}_2$ are disjoint.

It will turn out that we can construct symmetric 4-covers for the hypercubes Q_5, Q_6, Q_7 and Q_8. One way to this is to start with the block list $\mathcal{B}'$ of (5.9) and (5.8). For the reader's convenience, we represent the list once again.

$B'_1 = (1, 3, 5, 7),$
$B'_2 = (0, 2, 4, 6),$
$B'_3 = (0, 3, 5, 6),$
$B'_4 = (0, 3, 4, 7).$

Fig. 7.1

In Section 5.4 we mentioned that $\mathcal{B}'$ generates a symmetric snake $\mathcal{S}^{(8)}$ of length 2^6 in Q_8. Since the list $\mathcal{B}'$ is well-ordered, we can apply Corollary 6.5. Together with Corollary 6.4, this results in four disjoint snakes:

$$\mathcal{S}^{(8)}, \quad \mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_4, \quad \mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_5, \quad \mathcal{S}^{(8)} + \mathbf{e}_4 + \mathbf{e}_5.$$

Since these four snakes are all of length 2^6, they constitute a symmetric 4-cover of Q_8 (cf. also [19], where a similar result was obtained).

Next, we shall introduce the notion of *invariance group* of a cover of Q_n by snakes. Let

$$\mathcal{C}^{(n)} := \{ \mathcal{S}_1^{(n)}, \ \mathcal{S}_2^{(n)}, \ \ldots, \ \mathcal{S}_N^{(n)} \} \tag{7.1}$$

be a family of mutually disjoint snakes covering the whole vertex set VQ_n of the hypercube Q_n. We consider permutations of the elements of VQ_n, i.e. elements of the symmetric group S_{2^n} acting on the set of the 2^n vertices of Q_n. If π is such a permutation which has the property that $\{\mathbf{v}, \mathbf{w}\} \in EQ_n$ if and only if $\{\pi(\mathbf{v}), \pi(\mathbf{w})\} \in EQ_n$, for all $\mathbf{v}, \mathbf{w} \in VQ_n$, then π is called an *automorphism* of Q_n. All these automorphisms form a group called the *automorphism group* $Aut(Q_n)$ (cf. [3]).

It follows easily from the nearness condition (*) and the separability condition (**) (see Chapter 1), that a snake $\mathcal{S}$ is transformed into a snake $\pi(\mathcal{S})$, for any $\pi \in Aut(Q_n)$. Now we ask the question if there are $\pi \in Aut(Q_n)$ which transform $\mathcal{S}$ into $\mathcal{S}$ itself, or more generally, which $\pi \in Aut(Q_n)$ transform each snake $\mathcal{S}_i^{(n)}$ of the cover $\mathcal{C}^{(n)}$ into some snake $\mathcal{S}_j^{(n)}$ (the same snake or a different one) of $\mathcal{C}^{(n)}$. Of course, such elements π constitute a subgroup of $Aut(Q_n)$, which we shall call $Aut(\mathcal{C}^{(n)})$. So, we have the following definition.

Definition 7.1

The invariance group $Aut(\mathcal{C}^{(n)})$ of a cover $\mathcal{C}^{(n)}$ of Q_n is the group of those permutations of $Aut(Q_n)$ which induce a permutation of the snakes in $\mathcal{C}^{(n)}$.

Any subgroup of $Aut(\mathcal{C}^{(n)})$ will be called *an* invariance group of $\mathcal{C}^{(n)}$. It is well-known (and can be verified immediately) that the translations over vectors of $GF(2)^n$ define a subgroup of $Aut(Q_n)$ of order 2^n.

Now it will be clear that if we define in $GF(2)^8$ the translation group

$$\mathcal{K}^{(8)} = \{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_4, \mathbf{e}_0 + \mathbf{e}_5, \mathbf{e}_4 + \mathbf{e}_5\}, \tag{7.2}$$

each element of this group brings about a permutation of the vertices of Q_8 such that the cover

$$\mathcal{C}^{(8)} = \{\mathcal{S}^{(8)}, \mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_4, \mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_5, \mathcal{S}^{(8)} + \mathbf{e}_4 + \mathbf{e}_5\},$$

is invariant for that permutation. More precisely, $\mathcal{K}^{(8)}$ is isomorphic to an invariance group (permutation group) of $\mathcal{C}^{(8)}$. Therefore, we shall say that $\mathcal{K}^{(8)}$ itself is an invariance *translation* group of $\mathcal{C}^{(8)}$.

In Section 5.4, we demonstrated that by removing $\mathcal{B}_1$', and by puncturing with respect to coordinate 1, we obtain a snake $\mathcal{S}^{(7)}$ of length 2^5 in Q_7. If we also puncture the vectors of $\mathcal{K}^{(8)}$ with respect to 1, i.e. leaving out the components with label 1, we obtain a translation group $\mathcal{K}^{(7)}$ acting on VQ_7. Of course, as algebraic groups, $\mathcal{K}^{(8)}$ and $\mathcal{K}^{(7)}$ are isomorphic, since all omitted components were equal to 0.

By applying Corollaries 6.4 and 6.5, it follows that the family of snakes

$$\mathcal{C}^{(7)} = \{\mathcal{S}^{(7)}, \mathcal{S}^{(7)} + \mathbf{e}_0 + \mathbf{e}_4, \mathcal{S}^{(7)} + \mathbf{e}_0 + \mathbf{e}_5, \mathcal{S}^{(7)} + \mathbf{e}_4 + \mathbf{e}_5\} \tag{7.3}$$

is a symmetric 4-cover of Q_7 with an invariance translation group

$$\mathcal{K}^{(7)} = \{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_4, \mathbf{e}_0 + \mathbf{e}_5, \mathbf{e}_4 + \mathbf{e}\}.$$

Actually, the vectors $\mathbf{e}_0$, $\mathbf{e}_4$ and $\mathbf{e}_5$ in $\mathcal{C}^{(7)}$ and $\mathcal{K}^{(7)}$ differ from those in $\mathcal{C}^{(8)}$ and $\mathcal{K}^{(8)}$, since the first ones are in $GF(2)^7$ and the latter ones in $GF(2)^8$. However, to avoid unnecessary complex notations, we assume that this difference has been taken care of by the labels of $\mathcal{C}$ and $\mathcal{K}$.

Continuing in this way, i.e. first leaving out block B'_4 and puncturing with respect to 7, provides us with a snake $\mathcal{S}^{(6)}$ of length 2^4 and a symmetric 4-cover of Q_6. Finally, we leave out one of the remaining two blocks, say B'_3 and puncture with respect to 3, which provides us with a snake $\mathcal{S}^{(5)}$ of length 2^3, and consequently with a symmetric 4-cover of Q_5. So, all together, we constructed a series of four snakes

$$\mathcal{S}^{(8)}, \quad \mathcal{S}^{(7)}, \quad \mathcal{S}^{(6)}, \quad \mathcal{S}^{(5)}, \tag{7.4}$$

a series of isomorphic invariance translation groups

$$\mathcal{K}^{(8)}, \quad \mathcal{K}^{(7)}, \quad \mathcal{K}^{(6)}, \quad \mathcal{K}^{(5)} \tag{7.5}$$

and a series of covers

$$\mathcal{C}^{(i)} := \{\mathcal{S}^{(i)} + \mathcal{K}^{(i)}\} = \{\mathcal{S}^{(i)} + \mathbf{t} \mid \mathbf{t} \in \mathcal{K}^{(i)}\} \tag{7.6}$$

for $5 \leq i \leq 8$. The reason that we can apply Corollaries 6.4 and 6.5 every time (and that all these translation groups are isomorphic) is that the components of the translation vectors (7.2) are equal to 0 at the positions we puncture to, i.e. we do not puncture to coordinate 0, 4 and 5. We formulate the above construction in terms of a theorem.

Theorem 7.1

Let $\mathcal{S}^{(8)}$ *be the snake generated by the block list of* $\mathcal{B}$*'* *of* (5.9)*, and let* $\mathcal{S}^{(i)}$, $5 \le i \le 8$, *be the snakes of* (7.4)*. Let furthermore* $\mathcal{K}^{(i)}$, $5 \le i \le 8$, *be the series of isomorphic translation groups defined in* (7.5)*. Then the four snakes of the family* $\mathcal{C}^{(i)}$ *defined in* (7.6) *constitute a symmetric* 4*-cover of* Q_i *with* $\mathcal{K}^{(i)}$ *as invariance translation group, for every* i, $5 \le i \le 8$.

Example 7.1

Removing blocks B'_1 and block B'_2 from the block list (5.9) leaves us with the list

$$B'_2 = (0, 2, 4, 6),$$
$$B'_3 = (0, 3, 5, 6).$$

The transition sequence corresponding to this list is equal to

$$B'_2, B'_3, B'_2, B'_3 = \ 0, 2, 4, 6, 0, 3, 5, 6, 0, 2, 4, 6, 0, 3, 5, 6.$$

This sequence generates a snake $\mathcal{S}^{(6)}$ in Q_6 where the vertices are labeled 0, 2, 3, 4, 5 and 6. Translation of this snake over the vectors $\mathbf{e}_0 + \mathbf{e}_4$, $\mathbf{e}_0 + \mathbf{e}_5$ and $\mathbf{e}_4 + \mathbf{e}_5$ gives a family of four snakes which are mutually disjoint, and which form a 4-cover for Q_6. If we also leave out block B'_3, we are left with a single block B'_2 which defines the transition sequence

$$B'_2, B'_2 = 0, 2, 4, 6, 0, 2, 4, 6. \tag{7.7}$$

This sequence generates a snake $\mathcal{S}^{(5)}$ in Q_5 where the vertices are labeled 0, 2, 4, 5 and 6. Again translating $\mathcal{S}^{(5)}$ over the vectors $\mathbf{e}_0 + \mathbf{e}_4$, $\mathbf{e}_0 + \mathbf{e}_5$ and $\mathbf{e}_4 + \mathbf{e}_5$ provides us with a 4-cover of Q_5. For the explicit form of the above covers for Q_5 and Q_6 (also for Q_7 and Q_8), we refer to Appendix B.

We can also interpret (7.7) as the transition sequence of a snake $\mathcal{S}^{(4)}$ in Q_4, the vertices of which are labeled by 0, 2, 4 and 6. In this case, $\mathbf{e}_0 + \mathbf{e}_4$ is the only nonzero translation vector that transforms $\mathcal{S}^{(4)}$ into a disjoint snake. Stated equivalently, we now have a 2-cover for Q_4. Appendix B shows the explicit list of this 2-cover.

An easy way to obtain an 8-cover of Q_{15} is to start with the well-ordered list of Fig. 5.15 which generates a snake $\mathcal{S}^{(15)}$ of length 2^{12} in Q_{15}. Since it is well-ordered, we may conclude from Corollaries 6.4 and 6.5 that the following eight snakes are mutually disjoint:

$$\{\mathcal{S}^{(15)}, \quad \mathcal{S}^{(15)} + \mathbf{e}_0 + \mathbf{e}_{12}, \quad \mathcal{S}^{(15)} + \mathbf{e}_0 + \mathbf{e}_{13}, \quad \mathcal{S}^{(15)} + \mathbf{e}_0 + \mathbf{e}_{14}, \quad \mathcal{S}^{(15)} + \mathbf{e}_{12} + \mathbf{e}_{13},$$
$$\mathcal{S}^{(15)} + \mathbf{e}_{12} + \mathbf{e}_{14}, \quad \mathcal{S}^{(15)} + \mathbf{e}_{13} + \mathbf{e}_{14}, \quad \mathcal{S}^{(15)} + \mathbf{e}_0 + \mathbf{e}_{12} + \mathbf{e}_{13} + \mathbf{e}_{14}\} \tag{7.8}$$

More precisely, the set or family of snakes (7.8) constitutes a symmetric 8-cover of Q_{15}, since $\mathcal{S}^{(15)}$ and all its translations are symmetric (cf. (2.17)), and since these snakes together are incident with $2^3 \cdot 2^{12} = 2^{15}$ vertices of Q_{15}. We can phrase this slightly differently, by introducing the group of translations

$$\mathcal{H}^{(15)} = \{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_{12}, , \mathbf{e}_0 + \mathbf{e}_{13}, \mathbf{e}_0 + \mathbf{e}_{14}, \mathbf{e}_{12} + \mathbf{e}_{13}, \mathbf{e}_{12} + \mathbf{e}_{14}, \mathbf{e}_{13} + \mathbf{e}_{14}, \mathbf{e}_0 + \mathbf{e}_{12} + \mathbf{e}_{13} + \mathbf{e}_{14}\}, \tag{7.9}$$

and we can write the family (7.8) symbolically as

$$\mathcal{S}^{(15)} + \mathcal{H}^{(15)}. \tag{7.10}$$

So, the set (7.8) or (7.10) of eight snakes, which is a symmetric 8-cover of Q_{15}, has the translation group $\mathcal{H}^{(15)}$ of (7.9) as an *invariance translation group*.

As we discussed in Section 5.5, when removing block B_1 from the list of Fig. 5.15, we obtain a well-ordered list which generates a snake $\mathcal{S}^{(14)}$ of length 2^{11} in Q_{14}, after puncturing with respect to the coordinate with index 3. We also puncture the vectors of $\mathcal{H}^{(15)}$ with respect to this coordinate, providing us with a translation group $\mathcal{H}^{(14)}$. Actually, $\mathcal{H}^{(14)}$, as an (algebraic) group, is isomorphic with $\mathcal{H}^{(15)}$ and acts on the vertex set of a hypercube Q_{14}, which is isomorphic to a subgraph of Q_{15} (cf. Section 5.5 and the discussion w.r.t. $\mathcal{K}^{(8)}$ and $\mathcal{K}^{(7)}$ in the beginning of this Section)..

Like in Section 5.5 and with respect to the coordinate indices 2, 1, 4, 5, 8, we can continue this process of puncturing, thus obtaining a series of seven snakes

$$\mathcal{S}^{(15)}, \quad \mathcal{S}^{(14)}, \quad \mathcal{S}^{(13)}, \quad \mathcal{S}^{(12)}, \quad \mathcal{S}^{(11)}, \quad \mathcal{S}^{(10)}, \quad \mathcal{S}^{(9)} \tag{7.11}$$

and a series of isomorphic translation groups

$$\mathcal{H}^{(15)}, \quad \mathcal{H}^{(14)}, \quad \ldots, \quad \mathcal{H}^{(9)} \tag{7.12}$$

We also introduce the symbolic notation

$$\mathcal{C}^{(i)} := \{\mathcal{S}^{(i)} + \mathcal{H}^{(i)}\} = \{\mathcal{S}^{(i)} + \mathbf{t} \mid \mathbf{t} \in (\mathcal{H}^{(i)}, VQ_i)\} \tag{7.13}$$

for $9 \le i < 16$, as a generalization of (7.10).

It will be clear, using Corollaries 6.4 and 6.5, that the family of snakes $\mathcal{C}^{(i)} := \mathcal{S}^{(i)} + \mathcal{H}^{(i)}$ is a symmetric 8-cover of Q_i, for $9 \le i < 16$. For any of these i–values, we are entitled to apply Corollary 6.5, since the corresponding block list is well-ordered. We collect the above results in the following theorem.

Theorem 7.2

Let $\mathcal{S}^{(15)}$ be the snake generated by the block list of Fig. 5.15, *and let $\mathcal{S}^{(i)}$, $9 \le i < 16$, be the series of snakes of* (7.11). *Let furthermore $\mathcal{H}^{(i)}$, $9 \le i < 16$, be the series of isomorphic translation groups defined in* (7.12). *Then the eight snakes of the family $\mathcal{C}^{(i)}$ defined in* (7.13) *constitute a symmetric* 8*-cover of Q_i with $\mathcal{H}^{(i)}$ as invariance group for every i with* $9 \le i < 16$.

7.2 Near-Covers of Q_{16}

We apply Theorem 6.8 to the block list $\mathcal{B}$ in Fig. 5.19. First we take the pair (p_3, q_3) equal to (9, 10). It can rather easily be verified that there are no codewords $\mathbf{c}$ of the form (6.3) with sup $\mathbf{c}$ either of type $\{9, 10, i_4, j_4\}$ or of type $\{i_2, j_2, 9, 10\}$ (see next page). Together with the results of Corollary 6.4 for the pairs $\{0, 9\}$ and $\{0, 10\}$ and Theorem 6.8, we now have the following set of mutual disjoint snakes (*parallel snakes*).

$$\{\mathcal{S}, \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_9, \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{10}, \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{10}\}, \tag{7.14}$$

while the four translations

$$\{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_9, \mathbf{e}_0 + \mathbf{e}_{10}, \mathbf{e}_9 + \mathbf{e}_{10}\}$$

constitute a group with respect to binary addition. The union of the four snakes in (7.14) covers 2^{15} vertices of Q_{16}. In order to obtain a complete cover of Q_{16} we need more translations providing us with four more snakes which are mutual disjoint and also disjoint with those of the set (7.14).

We shall now explicitly verify that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{10} = \varnothing. \tag{7.15}$$

If (7.15) is not true, then according to Theorem 6.8, $\mathcal{C}$ contains a word $\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$ with support of the form $W = \{9, 10, i_4, j_4\}$ or $W = \{i_2, j_2, 9, 10\}$, with $4 < i < j$. The only possible cases for W are: $\{9, 10, 12, 15\}$, $\{9, 10, 13, 14\}$,$\{4, 7, 9, 10\}$ and $\{5, 6, 9, 10\}$. Here we used the fact that the Nim sum of the integers in a set W has to be equal to 0. If $W = \{4, 7, 9, 10\}$, it follows that $i = 5$, but $i_2 = 5 \notin W$. If $W = \{5, 6, 9, 10\}$, we have $i = 6$, $i_2 = 6$, $j_2 = 5$, and hence $j = 5$, which contradicts $i < j$.

If $W = \{9, 10, 13, 14\}$, we have $\mathbf{c} = \mathbf{b}_7 + \mathbf{b}_9$, and so $i = 8$, but $i_4 = 15 \notin W$. Finally, if $W = \{9, 10, 12, 15\}$, we have $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{11}$, $i = 7$, but $i_4 = 13 \notin W$. So in all cases we find a contradiction, and therefore (7.15) must hold.

In a completely similar way, we can prove

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{10} = \varnothing, \tag{7.16}$$

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{10} + \mathbf{e}_{11} = \varnothing, \tag{7.17}$$

Furthermore, Theorem 6.8 shows that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{11} \neq \varnothing, \tag{7.18}$$

since $\mathcal{C}$ contains a codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11}$ with support $W = \{9, 11, 13, 15\} = \{9, 11, 7_4, 8_4\}$ which is of the form (i), with $i = 7, j = 8$. Similarly, we have

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11} \neq \varnothing, \tag{7.19}$$

because of the existence of the codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_{10} + \mathbf{b}_{11}$ with $W = \{8, 11, 13, 14\} = \{8, 11, 7_4, 9_4\}$ which is of type (i), and

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_9 \neq \varnothing, \tag{7.20}$$

because of the existence of $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_9 + \mathbf{b}_{11}$, with $W = \{8, 9, 12, 13\} = \{8, 9, 7_4, 10_4\}$ (type (i)) and of $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_9$, with $W = \{6, 7, 8, 9\} = \{7_2, 11_2, 8, 9\}$ (type (ii)).

In order to investigate translations over vectors consisting of more than two unit vectors, we formulate two more theorems.

Theorem 7.3

Let $\mathcal{B}$ be the block list that satisfies the conditions of Theorem 6.8. *If the sequence $\overline{S}_k(\mathcal{B})$ of* (4.8) *generates a snake $\mathcal{S}$ and if p_3, q_3, r_3, s_3 are pairwise unequal, then we have*

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} + \mathbf{e}_{r_3} + \mathbf{e}_{s_3} = \varnothing,$$

if and only if there is no codeword $\mathbf{c} \in \mathcal{C}$, as expressed by (6.3), *with $W = sup\ \mathbf{c}$ equal to one of the following sets*

(i) $W = \{p_3, q_3, r_3, s_3, i_4, j_4\}$;

(ii) $W = \{i_2, j_2, p_3, q_3, r_3, s_3\}$,

for $m < i < j \le k$, and for any p and q with $m - 1 \le p, q \le k$, and $1 \le r, s, \le k$.

The proof is similar to the proof of Theorem 6.8.

We remark that, although $i \ne j$, the integers i_2 and j_2 may be equal to each other, as well as i_4 and j_4. Next we state the following generalization of Corollary 6.4.

Theorem 7.4

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a block list corresponding to an ordered minimum-weight basis of an $[n,k,4]$-code $\mathcal{C}$ which satisfies the fixed-position property. If the sequence $\overline{S}_k(\mathcal{B})$ of (4.8) *generates a snake $\mathcal{S}$, then*

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{b_1} + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} + \mathbf{e}_{r_3} = \varnothing$$

and

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_{b_2} + \mathbf{e}_{p_4} + \mathbf{e}_{q_4} + \mathbf{e}_{r_4} = \varnothing,$$

for $b, p, q, r \in \{1, 2, \ldots, k\}$.

The proof is identical to the proof of Corollary 6.4, and is based on Lemma 4.1.

Example 7.2

Consider again the block list $\overline{\mathcal{B}}$ of Fig. 5.19. The codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11}$ has a support $W = \{8, 9, 10, 11\}$ which is of type (*ii*) in Theorem 7.3 with $i = 7$ and $j = 8$, 9 or 10. Applying this theorem for $i_2 = j_2$ gives that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_9 + \mathbf{e}_{10} + \mathbf{e}_{11} \ne \varnothing \tag{7.21}$$

An application of Theorem 7.4 yields

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_i + \mathbf{e}_j + \mathbf{e}_k = \varnothing \tag{7.22}$$

for any choice of $i, j, k \in \{8, 9, 10, 11\}$.

The relations (7.15), (7.16), (7.17) and (7.22) illustrate that we can construct, in many ways, sets of four pairwise disjoint snakes, but there seems to be no such set of eight disjoint snakes of the same type as turned out to exist for Q_9, Q_{10}, …, Q_{15} (cf. Sect. 7.1). We conclude that the snake $\mathcal{S}$ generated by the block list of Fig. 5.19 does not give rise to a cover of Q_{16} if

we restrict ourselves to translations of $\mathcal{S}$ over vectors $\mathbf{e}_0 + \mathbf{e}_{p_3}$, $\mathbf{e}_0 + \mathbf{e}_{q_3}$, $\mathbf{e}_{p_3} + \mathbf{e}_{q_3}$ and $\mathbf{e}_0 + \mathbf{e}_{p_3} + \mathbf{e}_{q_3} + \mathbf{e}_{r_3}$.

However, there do exist sets consisting of eight snakes which 'almost' cover Q_{16}. If we extend e.g. the set (7.14) by the four snakes $\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8$, $\mathcal{S} + \mathbf{e}_8 + \mathbf{e}_9$, $\mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{10}$, $\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_9 + \mathbf{e}_{10}$, we end up with eight snakes with only four pairs of intersecting snakes, i.e. only the intersections $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_9$, $\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8 \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_9$, $\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{10} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_9 + \mathbf{e}_{10}$ and $\mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{10} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{10}$ are not empty.

We shall present a similar set of eight snakes together with the non-empty intersections themselves. Consider the following set

$$\begin{aligned} \{&\mathcal{S}, \quad \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8, \quad \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{10}, \quad \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{10}, \\ &\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{11}, \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}, \quad \mathcal{S} + \mathbf{e}_{10} + \mathbf{e}_{11}, \quad \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_{10} + \mathbf{e}_{11}\} \end{aligned} \tag{7.23}$$

From Corollary 6.4, and eqs. (7.16), (7.17), (7.19) and (7.22) we conclude that any two snakes of the set (7.23) are disjoint except the pairs $\{\mathcal{S}, \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}\}$, $\{\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8, \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{11}\}$, $\{\mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{10}, \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_{10} + \mathbf{e}_{11}\}$, $\{\mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{10}, \mathcal{S} + \mathbf{e}_{10} + \mathbf{e}_{11}\}$. By a computer calculation, we found

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11} = \{\mathbf{w}_{1023}, \mathbf{w}_{1279}, \mathbf{w}_{3071}, \mathbf{w}_{3327}, \mathbf{w}_{5119}, \mathbf{w}_{5375}, \mathbf{w}_{7167}, \mathbf{w}_{7423}\}. \tag{7.24}$$

In order to deal with the size of intersections of snakes like the one in (7.24), we now present the following property.

Theorem 7.5

Let $\overline{S}_k(\mathcal{B})$ be a transition sequence in Q_n, $n \geq 3$, generated by a basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k)$ of a linear $[n, k, 4]$-code $\mathcal{C}$ satisfying the conditions of Lemma 4.1. Let $\mathbf{c} \in \mathcal{C}$ be written as $\mathbf{c} = \sum_{l \in A} \mathbf{b}_l$, where A is some subset of $\{1, 2, \ldots, k\}$ and let $W := \sup \mathbf{c}$, $a := \min A$ and $i := a + 1$. Then the number of subsequences $\mathcal{T} := B_i, \mathcal{T}', B_j$, for some fixed $j > i$, with $c(\mathcal{T}) = W$ is equal to 2^{k-j}, for $j < k$, and to 2 for $j = k$.

Since $i > 1$, the proof follows immediately from Theorem 2.9 (*ii*). We shall apply the theorem to the intersection $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}$. The only reason for the non-emptiness of this intersection is the existence of the codeword $\mathbf{c}$ with support $W = \{8, 11, 13, 14\}$ which can be interpreted as $\{8, 11, i_4, j_4\}$ with $i = 7 < j = 9$, w.r.t. to the list of Fig. 5.19. More precisely, the size of the intersection is twice the number of occurrences of subsequences $\mathcal{T} := B_7, \mathcal{T}', B_9$ in $T_{11}(\mathcal{B})$ with $c(\mathcal{T}) = W$. It follows from Theorem 7.3 that this number is equal to $2^{11-9} = 4$, and so $|\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}| = 8$, which corresponds to the computer result (7.24).

To obtain the words in the intersection themselves we study the subsequences $(7, T', 9)$ in the transition sequence T_{11} of the standard Gray code $G(11)$:

6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•10•6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•11•
6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•10•6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•11•

In the above sequence, every dot '•' stands for $T'(5)$, the incomplete transition sequence of $G(5)$ of length 31.

Since $W = (8, 11, 13, 14) = B_6 \Delta B_7 \Delta B_{10} \Delta B_{11}$, this set is the support of the codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_{10} + \mathbf{b}_{11}$. Hence $\mathbf{c}' = \mathbf{b}_i + \mathbf{c} = \mathbf{b}_i + \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_{10} + \mathbf{b}_{11}$ (cf. the proof of Theorem 6.8). Therefore, we have to find those subsequences 7, T', 9 with $c(7, T', 9) = \{6, 10, 11\}$. Such a subsequence is

7•6•8•6•7•6•10•6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•11•6•7•6•8•6•7•6•9

of length 961. The first integer 7 has position $(9 + 10 \cdot 31) + 1 = 320$, and so the first block B_7 starts at position $4 \cdot 319 = 1276$ (remember the first word of $\mathcal{S}$ has index zero).

In the same way, we find that the final block B_9 of the subsequence B_7, $\mathcal{T}'$, B_9 ends at position $4 \cdot 1280 = 5120$. Applying the rules for the index-codeword conversion (cf. Section 5), we find that

$$\mathbf{w}_{1023} = \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{e}_{9_4}, \quad \mathbf{w}_{5375} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11} + \mathbf{e}_{7_4}$$

Hence, $\sup \mathbf{w}_{1023} = \{8, 9, 15\}$ and $\sup \mathbf{w}_{5375} = \{9, 11, 15\}$ and so we find that $\mathbf{w}_{1023} + \mathbf{e}_8 + \mathbf{e}_{11} = \mathbf{w}_{5375}$, which implies that $\mathbf{w}_{1023}$ and $\mathbf{w}_{5375}$ both belong to $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}$. In this way, we can verify that

$$\begin{aligned} \mathbf{w}_{1023} &= \mathbf{w}_{5375} + \mathbf{e}_8 + \mathbf{e}_{11}, \\ \mathbf{w}_{3071} &= \mathbf{w}_{7423} + \mathbf{e}_8 + \mathbf{e}_{11}, \\ \mathbf{w}_{5119} &= \mathbf{w}_{1279} + \mathbf{e}_8 + \mathbf{e}_{11}, \\ \mathbf{w}_{7167} &= \mathbf{w}_{3327} + \mathbf{e}_8 + \mathbf{e}_{11}. \end{aligned} \tag{7.25}$$

The three other non-empty intersections can be obtained by translating the set (7.24) over vectors $\mathbf{e}_0 + \mathbf{e}_8$, $\mathbf{e}_0 + \mathbf{e}_{10}$ and $\mathbf{e}_8 + \mathbf{e}_{10}$, respectively. The four intersections have no vector in common. E.g. suppose that $\mathbf{c} \in \mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_{11}$. Since $\mathbf{c} \in \mathcal{S}$, $\mathbf{c} + \mathbf{e}_0 + \mathbf{e}_8 \notin \mathcal{S}$ (cf. Corollary 6.4) and therefore $\mathbf{c} \notin \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_{11} \cap \mathcal{S} + \mathbf{e}_0 + \mathbf{e}_8$. It follows that there are $4 \cdot 8 = 32$ vertices of Q_{16} which are not incident with one of the eight snakes of (7.23). Since 32 is a relatively small number compared to 2^{16}, we could say that (7.23) provides us with a *near-cover* of Q_{16}.

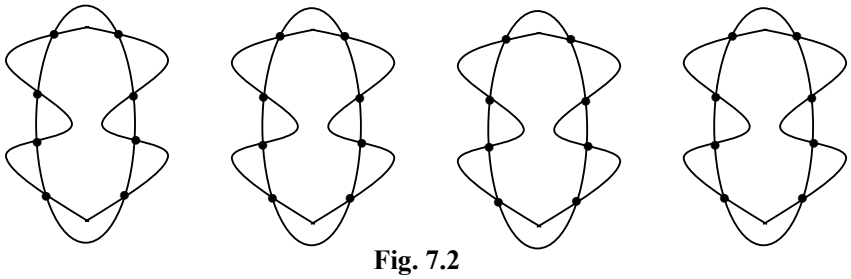

Fig. 7.2

The eight dots in the leftmost picture represent the eight words $\mathbf{w}_{1023}$, $\mathbf{w}_{5375}$, $\mathbf{w}_{3071}$, $\mathbf{w}_{7423}$, $\mathbf{w}_{5119}$, $\mathbf{w}_{1279}$, $\mathbf{w}_{7167}$ and $\mathbf{w}_{3327}$. The three other octets of dots represent the words obtained from the first

eight words by translating over $\mathbf{e}_0 + \mathbf{e}_8$, $\mathbf{e}_0 + \mathbf{e}_{10}$ and $\mathbf{e}_8 + \mathbf{e}_{11}$.

In a similar way, one can verify

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{11} = \{\mathbf{w}_{511}, \mathbf{w}_{2559}, \mathbf{w}_{4607}, \mathbf{w}_{6655}, \mathbf{w}_{5887}, \mathbf{w}_{7935}, \mathbf{w}_{1791}, \mathbf{w}_{3839}, \mathbf{w}_{5631}, \mathbf{w}_{7679}, \mathbf{w}_{1535}, \mathbf{w}_{3583}, \mathbf{w}_{767}, \mathbf{w}_{2815}, \mathbf{w}_{4863}, \mathbf{w}_{6911}\}, \tag{7.26}$$

with

$$\begin{aligned}
\mathbf{w}_{511} &= \mathbf{w}_{5887} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{2559} &= \mathbf{w}_{7935} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{4607} &= \mathbf{w}_{1791} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{6655} &= \mathbf{w}_{3839} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{5631} &= \mathbf{w}_{767} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{7679} &= \mathbf{w}_{2815} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{1535} &= \mathbf{w}_{4863} + \mathbf{e}_9 + \mathbf{e}_{11},\\
\mathbf{w}_{3583} &= \mathbf{w}_{6911} + \mathbf{e}_9 + \mathbf{e}_{11}.
\end{aligned} \tag{7.27}$$

So, $\mathcal{S} \cap \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{11}$ contains 16 words and hence by replacing $\mathbf{e}_8$ in (7.23) by $\mathbf{e}_9$, we obtain another near-cover of Q_{16} not covering 64 vertices. We can derive this number 16 as follows. The reason that this intersection is not empty is because of the existence of the codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11}$ with support $W = \{9, 11, 13, 15\}$, which is of type $\{9, 11, i_4, j_4\}$ with $i = 7 < 8 = j$. The number of times that a subsequence 7, T', 8 occurs in T_{11} with $c(7, T', 8) = \{6, 8, 9, 10, 11\}$ is equal to $2^{11-8} = 8.$, according to Theorem 7.5. Hence, we have $|\mathcal{S} \cap \mathcal{S} + \mathbf{e}_9 + \mathbf{e}_{11}| = 16$. The words in the intersection can be obtained in the same way as in the previous case. First we determine all subsequences 7, T', 8 with $c(7, T', 8) = \{6, 8, 9, 10, 11\}$. We found the following subsequences by inspection:

7•6•10•6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•11•6•7•6•8 (twice)

7•6•11•6•7•6•8•6•7•6•9•6•7•6•8•6•7•6•10•6•7•6•8 (twice)

7•6•9•6•7•6•8•6•7•6•10•…•11•6•7•6•8•6•7•6•9•6•7•6•8 (twice)

7•6•9•6•7•6•8•6•7•6•11•…•10•6•7•6•8•6•7•6•9•6•7•6•8 (twice)

The first four subsequences have length 705, whereas the second four have length 1217. E.g. in the first subsequence 7•6•10•…•11•6•7•6•8 the first integer has position 448, and hence the first integer in $B_7, B_6, B_{10}, \ldots, B_{11}, B_6, B_7, B_6, B_8$ has position $4 \cdot 447 = 1788$. Similarly, the last integer in B_8 has position 4608. This provides us with the words $\mathbf{w}_{1791}$ and $\mathbf{w}_{4607}$ which differ by $\mathbf{e}_9 + \mathbf{e}_{11}$.

We also find that

$$\mathcal{S} \cap \mathcal{S} + \mathbf{e}_8 + \mathbf{e}_9 = \{\mathbf{w}_{255}, \mathbf{w}_{2303}, \mathbf{w}_{4351}, \mathbf{w}_{6399}, \mathbf{w}_{2047}, \mathbf{w}_{4095}, \mathbf{w}_{6143}, \mathbf{w}_{8191}, \mathbf{w}_{2302}, \mathbf{w}_{6398}, \mathbf{w}_{4094}, \mathbf{w}_{8190}\}, \tag{7.28}$$

This intersection is not empty for three reasons. First, there exists a codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_9 + \mathbf{b}_{11}$ with support $W = \{8, 9, 12, 13\} = \{8, 9, i_4, j_4\}$ with $i = 7 < 10 = j$ (cf. Theorem 6.8 (i)). Secondly, this set can also be considered as $\{8, 9, i_4, j_4\}$ with $i = 7 < 11 = j$, which is also sufficient for the intersection being not empty (cf. Theorem 6.8(i)). In the third place, there exists a codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_9$ with support $W = \{6, 7, 8, 9\} = \{i_2, j_2, 8, 9\}$ with $i = 7 < 11 = j$.

From Theorem 7.3 we know that T_{11} contains two subsequences 7, T', 10 with contents {6, 7, 9, 11} and also two subsequences 7, T', 11 with the same contents. Finally, T_{11} contains two subsequences 7, T', 11 with contents {6, 9}. Altogether, the intersection contains $2\cdot(2 + 2 + 2) = 12$ words. In particular, we have

$$\begin{aligned} \mathbf{w}_{2047} &= \mathbf{w}_{4351} + \mathbf{e}_8 + \mathbf{e}_9, \\ \mathbf{w}_{4094} &= \mathbf{w}_{2302} + \mathbf{e}_8 + \mathbf{e}_9, \\ \mathbf{w}_{4095} &= \mathbf{w}_{2303} + \mathbf{e}_8 + \mathbf{e}_9, \\ \mathbf{w}_{6143} &= \mathbf{w}_{255} + \mathbf{e}_8 + \mathbf{e}_9, \\ \mathbf{w}_{8190} &= \mathbf{w}_{6398} + \mathbf{e}_8 + \mathbf{e}_9, \\ \mathbf{w}_{8191} &= \mathbf{w}_{6399} + \mathbf{e}_8 + \mathbf{e}_9. \end{aligned} \tag{7.29}$$

It follows that replacing $\mathbf{e}_{11}$ in (7.23) by $\mathbf{e}_9$ will yield a near-cover of size $2^{16} - 48$.

It appears that the words in the intersection (7.24) (and also in the intersections (7.26) and (7.28)) show a regular pattern with respect to their positions in the snake $\mathcal{S}$ and also with respect to their mutual dependency when considered as vectors in $GF(2)^{16}$. Let us consider e.g. the set (7.24) together with the relations (7.25). These relations are due to the existence of the codeword $\mathbf{c} = \mathbf{b}_6 + \mathbf{b}_7 + \mathbf{b}_{10} + \mathbf{b}_{11}$ with sup $\mathbf{c}$ = {8, 11, 13, 14}, or equivalently $\mathbf{c}' = \mathbf{c} + \mathbf{b}_7$ with sup $\mathbf{c}'$ = {0, 7, 10, 8, 11, 14} (cf. the first line right after (7.19)).

Actually, there are four subsequences $\overline{\mathcal{J}} = B_7, \mathcal{J}, B_9$ of $T_{11}(\overline{\mathcal{B}})$ with
$c(\overline{\mathcal{J}})$ = { 0, 7, 10, 8, 11, 14},
i.e.

$$[\mathbf{w}_{5372}, \mathbf{w}_{1024}], [\mathbf{w}_{7420}, \mathbf{w}_{3072}], [\mathbf{w}_{1276}, \mathbf{w}_{5120}] \text{ and } [\mathbf{w}_{3324}, \mathbf{w}_{7168}]. \tag{7.30}$$

Here, the notation

$$[\mathbf{w}_a, \mathbf{w}_b]$$

stands for a subsequence of $\mathcal{S}$ starting with $\mathbf{w}_a \in \mathcal{S}$ and ending with $\mathbf{w}_b \in \mathcal{S}$. Using $7_1 = 0$, $7_2 = 7$, $7_3 = 10$ and $9_4 = 14$, from (7.30) we obtain new subsequences

$$[\mathbf{w}_{5375}, \mathbf{w}_{1029}], [\mathbf{w}_{7423}, \mathbf{w}_{3071}], [\mathbf{w}_{1279}, \mathbf{w}_{5119}] \text{ and } [\mathbf{w}_{3327}, \mathbf{w}_{7167}], \tag{7.31}$$

which all have contents {8, 11}.

The existence of the four subsequences (7.31) can be explained as follows. According to Theorem 2.7, the transition sequence of the standard Gray code $G(15)$ certainly contains a subsequence $T_1 = 7, T', 9$ such that $c(T_1) = \{6, 10, 11\}$. By applying the symmetry relation of Theorem 2.4(*ii*), we obtain a second subsequence T_2 with $c(T_2) = c(T_1)$ at list distance 2^{11} from T_1. Here, we used Theorem 2.4(*ii*) that $c(T_1)$ is invariant for interchanging the integers 10 and 11.

By applying the symmetry relation of Theorem 2.4 (*i*), we find two more subsequences T_3 and T_4 with $c(T_3) = c(T_4) = c(T_1)$ at list distance 2^{12} from T_1 and T_2, respectively. This explains the constant difference of 2^{11} between the indices of the first words (and also of the last words) in the four subsequences of (7.30). Moreover, the same properties give rise to a linear dependency relation between those words. As one can easily verify, a shift over 2^{12}

positions in T_{11} corresponds to adding $\mathbf{b}_{10} + \mathbf{b}_{11}$ to all the codewords of $\mathcal{C}$, whereas interchanging 10 and 11 corresponds to an addition of $\mathbf{b}_9 + \mathbf{b}_{10}$.

Hence, we have

$$\begin{aligned} \mathbf{w}_{1024} &= \mathbf{b}_8 + \mathbf{b}_9, \\ \mathbf{w}_{3072} &= \mathbf{b}_8 \quad\quad + \mathbf{b}_{10}, \\ \mathbf{w}_{5120} &= \mathbf{b}_8 + \mathbf{b}_9 + \mathbf{b}_{10} + \mathbf{b}_{11}, \\ \mathbf{w}_{7168} &= \mathbf{b}_8 \quad\quad\quad\quad + \mathbf{b}_{11}, \end{aligned} \tag{7.32}$$

and so

$$\mathbf{w}_{1024} + \mathbf{w}_{3072} + \mathbf{w}_{5120} + \mathbf{w}_{7168} = \mathbf{0}. \tag{7.33}$$

Similar relations hold for the first and last words of the subsequences related to the intersections (7.26) and (7.28).

7.3 A Symmetric 8-Cover of Q_{16}

In this section we discuss snakes in Q_{16} constructed from a block list which corresponds to a basis of $R(2, 4)$ different from the basis in the previous sections.

Consider the list $\mathcal{B}$ of Fig. 5.18 consisting of eleven independent blocks satisfying the fixed position property, which corresponds to a basis of the Reed-Muller code $R(2, 4)$. For our convenience, we represent the same list here again.

$B_1 = (3, 5, 11, 13),$
$B_2 = (2, 5, 11, 12),$
$B_3 = (1, 5, 11, 15)$
$B_4 = (0, 5, 11, 14),$
$B_5 = (0, 6, 8, 14),$
$B_6 = (0, 6, 10, 12),$
$B_7 = (0, 7, 10, 13),$
$B_8 = (0, 4, 11, 15).$
$B_9 = (0, 7, 8, 15),$
$B_{10} = (0, 6, 9, 15),$
$B_{11} = (0, 6, 11, 13).$

Fig. 7.3

Just like all block lists in the previous chapter, the list in Fig 7.3 (or Fig. 5.18) also has the property that its sublist $(B_1, B_2, \ldots, B_{10})$ is well-ordered, and that adding B_{11} destroys that property. The only reason for this is that the integer $7_4 = 13$ now occurs in a block B_l with $l = 11 > 7$. Hence, we also get a well-ordered sublist when omitting block B_7.

Nevertheless, the list $\mathcal{B}$ of Fig 7.3 defines a transition sequence $T_{11}(\mathcal{B})$ which generates a snake of length 2^{13} in Q_{16}. Moreover, it will turn out that this snake can be translated over a number of eight translation vectors such that the result is a family of eight disjoint snakes.of length 2^{13} which constitutes a complete cover of Q_{16}.

Theorem 7.6

The transition sequence $T_{11}(\mathcal{B})$ determined by the block list of Fig. 7.3 *generates a snake $\boldsymbol{s}^{(16)}$ of length 2^{13} in Q_{16}.*

Proof. Since the list in Fig 7.3 is lexicographically ordered, we can apply Theorem 4.4, and verify that the underlying code $R(2, 4)$ does not contain words

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \ge i} \mathbf{b}_l$$

(cf. (6.3)) with a support W either of type (*i*) or of type (*ii*). Because list $\mathcal{B}$ is well-ordered if we leave out block B_7, we only have to check this condition for $i - 1 = 7$, or equivalently for $i = 8$. However, the only codewords of $R(2, 4)$ of type (*i*) or (*ii*) which are generated by the basis vectors $\mathbf{b}_7$, $\mathbf{b}_8$, $\mathbf{b}_9$, $\mathbf{b}_{10}$ and $\mathbf{b}_{11}$, are these basis vectors themselves, as one can verify immediately. So the only possible codeword

$$\mathbf{c} = \mathbf{b}_7 + \sum_{l \ge 8} \mathbf{b}_l$$

is $\mathbf{b}_7$.

Since in the block list of Fig 7.3 we have $7_2 \ne 8_2$ and $7_4 \ne 8_4$, $\mathbf{b}_7$ is not of type (*i*) or of type (*ii*). So, we are done. □

Theorem 7.7

Let $\mathcal{S}^{(16)}$ be the snake generated by the transition sequence $T_{11}(\mathcal{B})$, where $\mathcal{B}$ is the block list of Fig. 7.3. *Let furthermore $\mathcal{G}^{(16)}$ be the translation group of order* 8, *generated by the vectors*

$$\mathbf{e}_0 + \mathbf{e}_8, \qquad \mathbf{e}_0 + \mathbf{e}_9, \qquad \mathbf{e}_0 + \mathbf{e}_{11}.$$

Then the eight snakes of the set

$$\mathcal{C}^{(16)} = \{\mathcal{S}^{(16)} + \mathbf{t} \mid \mathbf{t} \in \mathcal{G}^{(16)}\}$$

constitute a symmetric 8*-cover of Q_{16} with invariance group $\mathcal{G}^{(16)}$.*

Proof. We can prove the Theorem along similar lines as demonstrated in the examples of the previous section, by applying Corollary 6.4 and Theorem 6.8. According to Corollary 6.4, we have that

$$\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_8 = \mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_9 = \mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_{11} = \varnothing.$$

Next we investigate the intersection $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_9$. This intersection is empty if and only if there is no codeword $\mathbf{c}$ expressed by (6.3) with a support W either of type $\{8, 9, i_4, j_4\}, j > i$, or of type $\{i_2, j_2, 8, 9\}, j > i$. We have the following possibilities for such words:

(a). $\{8, 9, 14, 15\} = B_5 \Delta B_{10}$;

(b). $\{8, 9, 12, 13\} = B_6 \Delta B_7 \Delta B_9 \Delta B_{10}$;

(c). $\{6, 7, 8, 9\} = B_9 \Delta B_{10}$;

(d). $\{4, 5, 8, 9\} = B_4 \Delta B_5 \Delta B_8 \Delta B_{10}$.

In case (*a*), it follows $i = 6$, but $14 \ne 6_4$ and $15 \ne 6_4$. In case (*b*), we have $i = 7$ and $13 = 7_4$, $12 = 6_4$. However $6 < 7$ $(= i)$ and so we have again a contradiction. In case (*c*) it follows $i = 10$, $6 = 10_2$, $7 = 9_2$, but $9 < 10$, while in case (*d*), $i = 5$, but $4 \ne 5_2$ and $5 \ne 5_2$. We conclude that $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_9 = \varnothing$.

Next we consider the possibility of codewords $\mathbf{c}$ expressed by (6.3) with a support W either of type $\{8, 11, i_4, j_4\}, j > i$, or of type $\{i_2, j_2, 8, 11\}, j > i$.

(e). $\{8, 11, 12, 15\} = B_6 \Delta B_7 \Delta B_9 \Delta B_{11}$, $i = 7$, $12 \neq 7_4$, $15 \neq 7_4$, a contradiction;

(f). $\{8, 11, 13, 14\} = B_5 \Delta B_{11}$, $i = 6$, $13 \neq 6_4$, $14 \neq 6_4$, a contradiction;

(g). $\{4, 7, 8, 11\} = B_8 \Delta B_9$, $i = 9$, $7 = 9_2$, $4 = 8_2$, $8 < 9$, a contradiction;

(h). $\{5, 6, 8, 11\} = B_4 \Delta B_5 \Delta B_8 \Delta B_9$, $i = 5$, $6 = 5_2$, $5 = 4_2$, $4 < 5$, a contradiction.

So, $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_{11} = \varnothing$.

Finally, we investigate the possibility of codewords $\mathbf{c}$ expressed by (6.3) with a support W either of type $\{9, 11, i_4, j_4\}, j > i$, or of type $\{i_2, j_2, 9, 11\}, j > i$.

(i). $\{9, 11, 12, 14\} = B_5 \Delta B_6 \Delta B_7 \Delta B_9 \Delta B_{10} \Delta B_{11}$, $i = 6$, $12 = 6_4$, $14 = 5_4$, $5 < 6$, a contradiction;

(j). $\{9, 11, 13, 15\} = B_{10} \Delta B_{11}$, $i = 11$, $13 = 11_4$, $15 = 10_4$, $10 < 11$, a contradiction;

(k). $\{4, 6, 9, 11\} = B_8 \Delta B_{10}$, $i = 9$, $4 \neq 9_2$, $6 \neq 9_2$, a contradiction;

(l). $\{5, 7, 9, 11\} = B_4 \Delta B_5 \Delta B_9 \Delta B_{10}$, $i = 5$, $5 \neq 5_2$, $7 \neq 5_2$, a contradiction.

We conclude that $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_9 + \mathbf{e}_{11} = \varnothing$.

Furthermore, by Theorem 7.4, we also have $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_9 + \mathbf{e}_{11} = \varnothing$. It is now a trivial task to show that the eight snakes

$$\mathcal{S}^{(16)}, \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_8, \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_9, \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_{11},$$
$$\mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_9, \mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_{11}, \mathcal{S}^{(16)} + \mathbf{e}_9 + \mathbf{e}_{11}, \mathcal{S}^{(16)} + \mathbf{e}_0 + \mathbf{e}_8 + \mathbf{e}_9 + \mathbf{e}_{11}$$

are pairwise disjoint. □

The property that the set $\mathcal{C}^{(16)}$ indeed covers Q_{16} was confirmed by a computer program. As we remarked already earlier we shall occasionally call a set $\mathcal{C}$ as described in the above theorem, a *complete family of parallel snakes*. Furthermore, we emphasize that Theorem 7.7 improves the result in [35, Section 4.3], where we presented a complete family of sixteen parallel symmetric snakes covering Q_{16} (a so-called *symmetric* 16-*cover*).

Remark 7.1

Similar investigations as above show that $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_{10} + \mathbf{e}_{11} = \varnothing$ and $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_8 + \mathbf{e}_{10} \neq \varnothing$, $\mathcal{S}^{(16)} \cap \mathcal{S}^{(16)} + \mathbf{e}_9 + \mathbf{e}_{10} \neq \varnothing$. Therefore, there are no more 8-covers of Q_{16} of the type described in Theorem 7.7 which are based on the list of Fig. 5.18.

In Section 7.1 we discussed a symmetric 8-cover of Q_{15}, and we demonstrated that it is possible – by successively leaving out blocks from the original block list in Fig. 5.15 and by puncturing to the relevant coordinates – to derive a series of related 8-covers for Q_{14}, Q_{13}, …, Q_9. It turns out that a similar hierarchy of covers exists when we start such a procedure from the 8-cover of Q_{16} as described in Theorem 7.7.

If we omit block B_1 in the list of Fig. 5.18, the remaining list generates a snake $\mathcal{S}^{(15)}$ of length 2^{12} in Q_{15} after puncturing with respect to the coordinate 3. Furthermore, this snake generates an 8-cover

$$\mathcal{C}^{(15)} = \{\mathcal{S}^{(15)} + \mathbf{t} \mid \mathbf{t} \in \mathcal{G}^{(15)}\},$$

where the group $\mathcal{G}^{(15)}$ is also generated by the vectors $\mathbf{e}_0 + \mathbf{e}_8$, $\mathbf{e}_0 + \mathbf{e}_9$, $\mathbf{e}_0 + \mathbf{e}_{11}$, but which are punctured with respect to coordinate 3. As one can verify rather easily, the proofs of these statements can be accomplished in the same way as the proofs of Theorem 7.6 and Theorem 7.7. Actually, they are identical, since B_1 and integer 3 did not play any role in the proofs of Theorem 7.6 and 7.7.

We continue this procedure by removing successively the blocks B_2, B_3, B_4, B_5, B_6 and B_9, and by puncturing with respect to 2, 1, 5, 14, 12 and 7, respectively. In this way, we obtain a series of eight symmetric snakes

$$\mathcal{S}^{(16)}, \quad \mathcal{S}^{(15)}, \quad \mathcal{S}^{(14)}, \quad \mathcal{S}^{(13)}, \quad \mathcal{S}^{(12)}, \quad \mathcal{S}^{(11)}, \quad \mathcal{S}^{(10)}, \quad \mathcal{S}^{(9)} \tag{7.34}$$

and a series of isomorphic translation groups

$$\mathcal{G}^{(16)}, \quad \mathcal{G}^{(15)}, \quad \ldots, \quad \mathcal{G}^{(9)}, \tag{7.35}$$

(cf. Theorem 7.7), and where $\mathcal{G}^{(i)}$, $9 \le i < 16$ are obtained from $\mathcal{G}^{(16)}$ by puncturing to the relevant coordinates as mentioned above.

Theorem 7.8

Let $\mathcal{S}$ be the snake generated by the block list of Fig. 7.3 *and let $\mathcal{S}^{(i)}$, $9 \le i \le 16$, be the series of snakes of* (7.34). *Let furthermore $\mathcal{G}^{(i)}$, $9 \le i \le 16$, be the series of isomorphic translation groups defined in* (7.35). *Then the family $\mathcal{C}^{(i)}$ of eight snakes*

$$\mathcal{C}^{(i)} := \{\mathcal{S}^{(i)} + \mathcal{G}^{(i)}\} = \{\mathcal{S}^{(i)} + \mathbf{t} \mid \mathbf{t} \in \mathcal{G}^{(i)}\}$$

constitutes a symmetric 8*-cover of Q_i with $\mathcal{G}^{(i)}$ as invariance translation group, for every i, $9 \le i \le 16$.*

The proof only slightly differs from the proofs of Theorem 7.6 and 7.7.

8. More about Covers of Hypercubes

In Section 8.1 of this chapter, we first introduce a result of Logacev concerning the Griesmer bound that gives a lower bound for the word length n of a linear $[n, k, d]$-code when k and d are fixed, and that gives an upper bound for k when n and d are fixed. We use this to obtain a lower bound for the number of snakes in covers of Q_n produced by our method.

In Section 8.2 we discuss sufficient conditions for extending a block list $\mathcal{B}$ generating a snake to a block list $\mathcal{B}^{ext}$ generating a complete Gray code which can be transformed quite easily into a cover of Q_n with snakes. Section 8.3 gives a general form of a block list corresponding to a basis of the Reed-Muller code $R(m-2, m)$ that generates a snake. This list can be extended to a block list generating a complete Gray code of word length $n = 2^m$, which in its turn provides us with a symmetric 2^{m-1}-cover.

8.1 Covers of Q_n and the Griesmer Bound

In the previous chapter we proved that for the hypercubes Q_n, $9 \le n \le 16$, there exist covers consisting of eight symmetric snakes (symmetric 8-covers) with an invariance group of order 8. In Section 7.1 we proved that for $5 \le n \le 8$, the hypercubes Q_n have symmetric 4-covers (cf. [19] and [33 - 36]), with an invariance group of order 4, and we noticed that Q_3 and Q_4 both have a 2-cover with an invariance group of order 2.

We also remarked in 7.1 that a 2-cover of Q_4 can be obtained by translating the snake generated by the transition sequence 0, 1, 2, 3, 0, 1, 2, 3 (cf. Section 5.4) over the vector $\mathbf{e}_0 + \mathbf{e}_2$. A 2-cover of Q_3 is obtained by doubling the 1-cover of Q_2, i.e. by taking the union of the two snakes covering two disjoint Q_2-subcubes of Q_3. It is trivial that Q_2 has a 1-cover, since the covering snake coincides with the graph itself.

It can easily be proved that for $2 \le n \le 8$, these results can not be improved. E.g. a 2-cover for Q_5 would consist of two snakes of length 2^4 whereas the longest snakes possible in Q_5 has length $s(5) = 14$ (e.g. [19, Section 1.3]). Similarly, a 2-cover for Q_8 would consist of two snakes of length 2^7, whereas $s(8) \le 123$ (cf. eq. (A.1), Appendix A).

As for a possible improvement of Theorem 7.8, we make the following observations. All covers in this thesis until now are based on linear $[n, k, 4]$-codes. The number of snakes in such a cover of Q_n is equal to 2^{n-k-2}. An obvious question is to ask for the minimal value of 2^{n-k-2} or, equivalently, for the minimal value of $n - k$, for fixed n.

From the well-known Griesmer bound for linear $[n,k,d]$-codes (cf. e.g. [23, Ch. 17])

$$n \ge g(k,d) = \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil, \tag{8.1}$$

it follows that for linear codes with minimum distance 4, the word length n satisfies

$$n \geq g(k,4) = 4 + 2 + (k - 2) = k + 4,$$

and consequently,

$$n - k - 2 \geq 2.$$

Hence, the minimal value for $n - k$ is 4, and our method only yields a 4-cover for Q_n if n meets the Griesmer bound.

A result of Logacev in [22] says for any linear $[n,k,d]$-code, that if

$$2 < d < 2^{k-2} - 1, \tag{8.2}$$

then

$$n \geq g(k,d) + 1. \tag{8.3}$$

It follows that for $d = 4$

$$n \geq k + 5,$$

as soon as $k \geq 5$.

Therefore, for $n \geq 9$, there is no linear $[n,k,4]$-code meeting the Griesmer bound. This implies that for $n \geq 9$, the best covers that can be constructed by our method, based on a minimum-distance-4 code, will be 8-covers.

Theorem 7.8 shows that for $n = 9, 10, \ldots, 16$, such covers really can be produced. The underlying linear codes, i.e. the Reed-Muller code $R(2, 4) \approx \mathcal{H}_4^{ext}$ with parameters [16, 11, 4], and the linear $[16 - i, 11 - i, 4]$-codes, $0 < i \leq 7$, obtained from $R(2, 4)$ by puncturing and omitting basis vectors, all satisfy the relation (8.3).

Proceeding in this way, we may conclude that for $16 < n \leq 32$, the best symmetric covers for Q_n (i.e. having a minimal number of snakes) that can be constructed by our method, starting with a minimal-distance-4 code, will be 16-covers. This is because $R(3, 5)$ is a [32, 26, 4]-code, and so $n - k - 2 = 4$, yielding that the minimal number of snakes in a cover of Q_{32} is $2^{n-k-2} = 16$. The only thing one has to prove to turn this conjecture into a theorem is that $R(3, 5)$ has an appropriate basis corresponding to a block list consisting of 26 blocks which satisfy the fixed-position property and which can be ordered such that the theorems of Chapters 4, 5 and 6 can be applied. The 16-covers for $Q_{31}, Q_{30}, \ldots, Q_{17}$ then will be obtained by the process of puncturing just like the 8-covers for $Q_{15}, Q_{14}, \ldots, Q_9$.

An alternative approach would be as follows. The 2-cover of Q_4 is the first cover which can be constructed with the method of this thesis starting with a minimum-distance-4 linear code. More precisely, this code is a [4, 1, 4]-code (cf. Section 7.1). By the process of doubling, i.e. by taking two disjoint Q_4-subgraphs in Q_5, we obtain a 4-cover for Q_5. The underlying linear code of this cover is a [5, 1, 4]-code. By properly extending the basis $\{\mathbf{b}_1\}$ of this code by vectors $\mathbf{b}_2$, $\mathbf{b}_3$ and $\mathbf{b}_4$ one by one, we can obtain bases of [6, 2, 4]-, [7, 3, 4]- and [8, 4, 4]-

codes and corresponding block lists which generate snakes and 4-covers for Q_6, Q_7 and Q_8, respectively.

When doubling the 4-cover of Q_8, we obtain a symmetric 8-cover of Q_9 with an underlying [9, 4, 4]-code. This cover, in its turn, can be extended to 8-covers for Q_{10}, Q_{11}, …, Q_{16} and corresponding codes with parameters [10, 5, 4], [11, 6, 4], …, [16, 11, 4]. The next series of 16-covers then could be obtained by doubling in an appropriate way the 8-cover of Q_{16} to a 16-cover for Q_{17} and to a basis of a [17, 11, 4]-code. However, at this moment, we do not (yet) have conditions by which we can decide when a basis is extendable such that the extended block list generates a snake and next a cover, and neither do we have an algorithm to accomplish such an extension in a systematic way.

Example 8.1

Consider the 2-cover of Q_4 as shown in Appendix B and extend the words (vectors) in the two lists by one extra coordinate. Although we can insert this extra coordinate on any position, we put it between the third and fourth coordinate and we label the five coordinate positions in the extended words by 0, 2, 4, 5 and 6. Of course, this labeling is also completely arbitrary and 0, 1, 2, 3, 4 would have been more natural, but we want to have the formulation as close as possible to Example 7.1.

So, we have the following two disjoint snakes in Q_5.

$\mathcal{S}^{(5)}$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_4$
0 0 0 0 0	1 0 1 0 0
1 0 0 0 0	0 0 1 0 0
1 1 0 0 0	0 1 1 0 0
1 1 1 0 0	0 1 0 0 0
1 1 1 0 1	0 1 0 0 1
0 1 1 0 1	1 1 0 0 1
0 0 1 0 1	1 0 0 0 1
0 0 0 0 1	1 0 1 0 1
Position Labels: 0 2 4 5 6	0 2 4 5 6

Now, the process of doubling provides us with another pair of disjoint snakes which are also disjoint with the first two.

$\mathcal{S}^{(5)} + \mathbf{e}_5$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_4 + \mathbf{e}_5$
0 0 0 1 0	1 0 1 1 0
1 0 0 1 0	0 0 1 1 0
1 1 0 1 0	0 1 1 1 0
1 1 1 1 0	0 1 0 1 0
1 1 1 1 1	0 1 0 1 1
0 1 1 1 1	1 1 0 1 1
0 0 1 1 1	1 0 0 1 1
0 0 0 1 1	1 0 1 1 1
Position Labels: 0 2 4 5 6	0 2 4 5 6

So, we end up with a 4-cover of Q_5. We emphasize that the invariance group of this cover is the translation group

$$\{\mathbf{0},\quad \mathbf{e}_0+\mathbf{e}_4,\quad \mathbf{e}_0+\mathbf{e}_4+\mathbf{e}_5,\quad \mathbf{e}_5\}.$$

which differs from the group

$$\{\mathbf{0},\quad \mathbf{e}_0+\mathbf{e}_4,\quad \mathbf{e}_4+\mathbf{e}_5,\quad \mathbf{e}_0+\mathbf{e}_5\},$$

the invariance group of the 4-cover in Appendix B.

8.2 Covers of Q_n and Gray Codes

It turns out that there is a relationship between the covers of Q_n, $3 \le n \le 16$, which we constructed in the previous sections, and Gray codes. More precisely, the various snakes of a particular cover of Q_n for some fixed value of n can be connected to each other via a small alteration such that the result is a complete cyclic Gray code in the very same hypercube.

As an example, we consider the cover of Q_8 which is explicitly shown in Appendix B. This cover consists of the following four snakes

$$\mathcal{S}^{(8)},\qquad \mathcal{S}^{(8)}+\mathbf{e}_0+\mathbf{e}_4,\qquad \mathcal{S}^{(8)}+\mathbf{e}_0+\mathbf{e}_5,\qquad \mathcal{S}^{(8)}+\mathbf{e}_4+\mathbf{e}_5, \tag{8.4}$$

where the snake $\mathcal{S}^{(8)}$ is generated by the block list

$$B_1=(1,3,5,7),\quad B_2=(0,2,4,6),\quad B_3=(0,3,5,6),\quad B_4=(0,3,4,7), \tag{8.5}$$

These four blocks correspond to a basis of an [8, 4, 4]-code $\mathcal{C}$.

We now extend this list with blocks

$$B_5=(0,3,0,7),\quad B_6=(0,3,5,7). \tag{8.6}$$

If we let block B_5 correspond to a vector $\mathbf{b}_5$ with $\sup \mathbf{b}_5 = \{3, 7\}$, and B_6 – as usual – to a vector $\mathbf{b}_6$ with $\sup \mathbf{b}_6 = \{0, 3, 5, 7\}$, we have an ordered basis $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4, \mathbf{b}_5, \mathbf{b}_6)$ which generates an [8, 6, 2]-code $\mathcal{C}^{ext}$. Now, the extended block list

$$\mathcal{B}^{ext}=(B_1,B_2,B_3,B_4,B_5,B_6), \tag{8.7}$$

is dealt with in the same way as all block lists in the previous sections which corresponded to $[n, k, 4]$-codes. The resulting sequence (cf. (4.8))

$$\overline{S}_6(\mathcal{B}^{ext})=B_1\,B_2\,B_1\,B_3\,\ldots\,B_1\,B_2\,B_1\,B_6\,B_1\,B_2\,B_1\,B_3\,\ldots\,B_1\,B_2\,B_1\,B_6 \tag{8.8}$$

turns out to be the transition sequence of a complete Gray code in Q_8. We can prove this by similar arguments as were used in part A of Theorem 4.2. It is sufficient to show that any proper subsequence of (8.8) contains at least one integer that occurs an odd number of times in that subsequence. Right after Example 8.2, we shall prove a general theorem dealing with this problem.

Now consider an arbitrary subsequence $\mathcal{T}$ of $\overline{S}_6(\mathcal{B}^{ext})$. We can always write this subsequence as $\mathcal{T}=\mathcal{T}_1\,\mathcal{T}_2\,\mathcal{T}_3$, where $\mathcal{T}_2$ consists of complete blocks, $0 \le |\mathcal{T}_1| \le 3$ and $0 \le |\mathcal{T}_3| \le 3$. Since the contents of $\mathcal{T}_2$ (cf. Definition 2.1) corresponds to a codeword of $\mathcal{C}^{ext}$ the number of integers occurring in $\mathcal{T}_2$ an odd number of times is at least 2, because the minimum distance

of $\mathcal{C}^{ext}$ is 2. By applying parity considerations w.r.t. the number of integers of the sets $P_1 = I_1 \cup I_3$ and $P_2 = I_2 \cup I_4$, one can easily understand that there is always at least one integer occurring an odd number of times in $\mathcal{T}$, unless $\mathcal{T}$ is the complete sequence $\overline{S}_6(\mathcal{B}^{ext})$. Since $|\overline{S}_6(\mathcal{B}^{ext})|$ is equal to $2^2 \cdot 2^6 = 2^8$, this proves that (8.8) defines a complete Gray code in Q_8.

Because of the properties of the standard Gray code, we can write (8.8) in the form

$$\overline{S}_6(\mathcal{B}^{ext}) = S_6(\mathcal{B}^{ext}), B_5, S_6(\mathcal{B}^{ext}), B_6, S_6(\mathcal{B}^{ext}), B_5, S_6(\mathcal{B}^{ext}), B_6$$

where $S_6(\mathcal{B}^{ext})$ stands for the transition sequence obtained form $\overline{S}_6(\mathcal{B}^{ext})$ by removing the last block of $\overline{S}_6(\mathcal{B}^{ext})$. If we define

$$S_6(\mathcal{B}^{ext}), B_5 := \mathcal{L}, 0, 7$$

and

$$S_6(\mathcal{B}^{ext}), B_6 := \mathcal{L}, 5, 7,$$

where

$$\mathcal{L} := S_6(\mathcal{B}^{ext}), 0, 3,$$

then the transition sequence $\overline{S}_6(\mathcal{B}^{ext})$ can also be written as

$$\overline{S}_6(\mathcal{B}^{ext}) = \mathcal{L}, 0, 7, \mathcal{L}, 5, 7, \mathcal{L}, 0, 7, \mathcal{L}, 5, 7. \tag{8.9}$$

So, we can interpret $\overline{S}_6(\mathcal{B}^{ext})$ as a *symmetric transition sequence* which is a concatenation of four transition sequences of the open snakes $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, while each of them is generated by either $\mathcal{L}, 0, 7$ or $\mathcal{L}, 5, 7$.

This is an example of a complete Gray code other than the standard Gray code. It is generated by a symmetric transition sequence, and moreover it is a concatenation of four disjoint open snakes. Although two of the four open snakes, $\mathcal{S}_1$ and $\mathcal{S}_3$, and also $\mathcal{S}_2$ and $\mathcal{S}_4$, are generated by the same transition sequence, the four snakes $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, are initialized by different words

$$\mathbf{s}_0 = \mathbf{0}, \quad \mathbf{s}_1 = \mathbf{b}_4 + \mathbf{b}_5, \quad \mathbf{s}_2 = \mathbf{b}_5 + \mathbf{b}_6, \quad \mathbf{s}_4 = \mathbf{b}_4 + \mathbf{b}_6$$

respectively. In fact these four initial words are the elements of a translation group

$$\mathcal{G} = <\mathbf{b}_4 + \mathbf{b}_5, \mathbf{b}_4 + \mathbf{b}_6> = \{\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_4\},$$

By interpreting the code $\mathcal{C}$ and

$$\mathcal{V} := \{\mathcal{C} \cup \mathcal{C} + \mathbf{s}_1 \cup \mathcal{C} + \mathbf{s}_2 \cup \mathcal{C} + \mathbf{s}_3\}$$

as algebraic groups, $\mathcal{G}$ is equivalent to the factor group $\mathcal{V}/\mathcal{C}$. The four cosets

$$\mathcal{C}, \qquad \mathcal{C} + \mathbf{s}_1, \qquad \mathcal{C} + \mathbf{s}_2, \qquad \mathcal{C} + \mathbf{s}_3,$$

are the 'skeletons' of the four open snakes $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, respectively.

The above interpretation leads to the question of the possibility to invert this process. Is it possible to introduce an ordered basis of an $[n, k', 2]$-code, with $k' > k$, which generates a Gray code in Q_n consisting of 2^a open snakes of length 2^{n-a}? If so, one only would have to

partition this Gray code into these 2^a open snakes and to construct for each such open snake an appropriate head-tail connection in order to end up with 2^a disjoint cyclic snakes covering Q_n.

In case of the above example, this connection is realized by a cyclic permutation of the last words of the four open snakes as one can verify in Appendix C. Before discussing the general theorem about this question, we present one more example.

Example 8.2

Consider the block lists

$$\mathcal{B} = (B_1)$$

and

$$\mathcal{B}^{ext} = (B_1, B_2, B_3),$$

with $B_1 = (0, 1, 2, 4)$, $B_2 = (0, 1, 0, 4)$ and $B_3 = (0, 1, 3, 4)$. The corresponding vectors $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3 \in GF(2)^5$ have supports $\{0, 1, 2, 4\}$, $\{1, 4\}$ and $\{0, 1, 3, 4\}$ and generate a [5, 3, 2]-code. It can easily be proved that

$$\overline{S}_3(\mathcal{B}^{ext}) = B_1\, B_2\, B_1\, B_3\, B_1\, B_2\, B_1\, B_3$$

generates a Gray code of word length 5. By substitution of the blocks, we obtain the following sequence

$$\overline{S}_3(\mathcal{B}^{ext}) = 0, 1, 2, 4, 0, 1, 0, 4, 0, 1, 2, 4, 0, 1, 3, 4, 0, 1, 2, 4, 0, 1, 0, 4, 0, 1, 2, 4, 0, 1, 3, 4.$$

The sequence can be written as

$$\overline{S}_3(\mathcal{B}^{ext}) = \mathcal{L}, 0, 4, \mathcal{L}, 3, 4, \mathcal{L}, 0, 4, \mathcal{L}, 3, 4;$$

where $\mathcal{L} = 0, 1, 2, 4, 0, 1$. Here,

$$\mathcal{L}, 0$$

and

$$\mathcal{L}, 3$$

stand for the transition sequences of four consecutive open snakes which are generated by $\overline{S}_3(\mathcal{B}^{ext})$ and which are translations of each other apart from their last words. The Gray code defined by $\overline{S}_3(\mathcal{B}^{ext})$ is presented by the following list.

0. 00000 (0)	8. 10100 (0)	16. 10010 (0)	24. 00110 (0)
1. 10000 (1)	9. 00100 (1)	17. 00010 (1)	25. 10110 (1)
2. 11000 (2)	10. 01100 (2)	18. 01010 (2)	26. 11110 (2)
3. 11100 (4)	11. 01000 (4)	19. 01110 (4)	27. 11010 (4)
4. 11101 (0)	12. 01001 (0)	20. 01111 (0)	28. 11011 (0)
5. 01101 (1)	13. 11001 (1)	21. 11111 (1)	29. 01011 (1)
6. 00101 (0)	14. 10001 (3)	22. 10111 (0)	30. 00011 (3)
7. 10101 (4)	15. 10011 (4)	23. 00111 (4)	31. 00001 (4)

By a cyclic permutation of the words with indices 7, 15, 23, and 31, we obtain the four snakes

$$\mathcal{S}^{(4)}, \quad \mathcal{S}^{(4)} + \mathbf{b}_1 + \mathbf{b}_2, \quad \mathcal{S}^{(4)} + \mathbf{b}_2 + \mathbf{b}_3, \quad \mathcal{S}^{(4)} + \mathbf{b}_1 + \mathbf{b}_3.$$

Since. $\mathbf{b}_1 + \mathbf{b}_2 = \mathbf{e}_0 + \mathbf{e}_2$, $\mathbf{b}_2 + \mathbf{b}_2 = \mathbf{e}_0 + \mathbf{e}_3$ and $\mathbf{b}_1 + \mathbf{b}_3 = \mathbf{e}_2 + \mathbf{e}_3$, this cover of Q_5 is identical to the cover which was derived at the end of Appendix B.

In general, we shall extend a lexicographically ordered block list $\mathcal{B}$ which corresponds to the basis of an $[n, k, 4]$-code, and which satisfies the fixed-position property, with $a := n - k - 2$ additional independent blocks $B_{k+1}, B_{k+2}, \ldots, B_{k+a}$, where

$$B_{k+1} = (0, k_2, 0, k_4) \tag{8.10}$$

and

$$B_l = (0, k_2, l_3, k_4), \tag{8.11}$$

for $k + 2 \le l \le k + a$, and $l_3 \in I_3$.

The blocks of (8.11) correspond to vectors $\mathbf{b}_l \in GF(2)^n$ with $\sup \mathbf{b}_l = \{0, k_2, l_3, k_4\}$, whereas we let B_{k+1} correspond to a vector $\mathbf{b}_{k+1}$ of weight 2 with $\sup \mathbf{b}_{k+1} = \{k_2, k_4\}$. So, the new block list $\mathcal{B}^{ext}$ consists of $k + a$ independent blocks, and is still lexicographically ordered since $(k + 1)_1 = (k + 2)_1 = \ldots = (k + a)_1 = 0$ (cf. 4.2(i)). However, it no longer satisfies the fixed-position property because of block B_{k+1}, and neither do the new blocks have a place in the underlying Euclidean Geometry of the original list $\mathcal{B}$, since the Nim sum of the integers in a block is unequal to 0. It is obvious that $\mathcal{B}^{ext}$ generates an $[n, k + a, 2]$-code, which we shall call $\mathcal{C}^{ext}$ since it contains $\mathcal{C}$ as a subcode.

The transition sequence $\overline{S}_{k+a}(\mathcal{B}^{ext})$ obtained by substituting the blocks of $\mathcal{B}^{ext}$ in (4.8) will, in general, not define a snake. This is because the minimum distance 2 of $\mathcal{C}^{ext}$ prevents us from proving the separability condition of a snake like we did in part B of the proof of Theorem 4.2. However, we can rather easily prove the validity of the nearness condition by assuming one additional (weak) condition for the blocks, and next slightly generalizing part A of that proof.

Theorem 8.1

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a lexicographically ordered block list corresponding to an $[n, k, 4]$-code satisfying the fixed-position property. Let $\mathcal{B}^{ext} = (B_1, B_2, \ldots, B_k, B_{k+1}, \ldots, B_{k+a})$ be the extended block list such that $B_{k+1}, \ldots, B_{k+a}$ are defined by (8.10) *and* (8.11). *Then $\mathcal{B}^{ext}$ is the transition sequence of a complete Gray code, if each B_i, $i \in \{1, 2, \ldots, k, k + 2, \ldots, k + a\}$ contains at least one integer that does not occur in any of the blocks B_l, $l > i$.*

Proof. We shall prove that all $2^2 \cdot 2^{k+a} = 2^n$ words of the list $\mathcal{S}$ generated by $\overline{S}_{k+a}(\mathcal{B}^{ext})$, starting from the zeroword, are different. The arguments are similar to those used in the proof of part A of Theorem 4.2. Take two different words $\mathbf{x}$ and $\mathbf{y}$ from this list $\mathcal{S}$. Just like in the proof of the above mentioned theorem, we can write w.l.o.g., $\mathbf{x} = \mathbf{c}' + \mathbf{z}'$ and $\mathbf{y} = \mathbf{c}'' + \mathbf{z}''$, where $\mathbf{c}', \mathbf{c}'' \in \mathcal{C}^{ext}$ and where $\mathbf{z}'$ is one of the vectors $\mathbf{0}$, $\mathbf{e}_{i_1}$, $\mathbf{e}_{i_1} + \mathbf{e}_{i_2}$, $\mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3}$, and $\mathbf{z}''$ is one of the vectors $\mathbf{0}$, $\mathbf{e}_{j_4}$, $\mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $\mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$, $1 \le i, j \le k + a$, where we, sloppily, allow i_3 and j_3 to

take on the value 0 in case $i = k + 1$ or $j = k + 1$ (cf. Fig. 4.1). We also partition $\overline{S}_{k+a}(\mathcal{B}^{ext})$ for fixed values i and j according to $\overline{S}_{k+a}(\mathcal{B}^{ext}) = \mathcal{T}'', B_i, \mathcal{T}', B_j, \mathcal{T}'''$.

Assume that $d(\mathbf{x}, \mathbf{y}) = 0$. W.l.o.g. we may conclude that $\mathcal{T}'$ and $\mathcal{T}'''$, $\mathcal{T}''$ are not empty. It follows that $\mathbf{c} := \mathbf{c}' + \mathbf{c}'' = \mathbf{z}' + \mathbf{z}''$ and $\mathbf{c} \neq \mathbf{0}$. Hence, because $\mathbf{c} \in \mathcal{C}^{ext}$, $\|\mathbf{c}\|$ is equal to 2, 4 or 6. From the form of the basis vectors, it follows that codewords of $\mathcal{C}^{ext}$ of weight 2 have a support which is either of type $\{i_b, j_b\}$, $b \in \{1, 2, 3, 4\}$ or of type $\{i_1, j_3\}$, or of type $\{i_2, j_4\}$, $1 \leq i, j \leq k + a$.

It is obvious that for sup $\mathbf{c} = \{i_1, j_3\}$, there are no $\mathbf{z}'$ and $\mathbf{z}''$ such that $\mathbf{c} + \mathbf{z}' + \mathbf{z}'' = \mathbf{0}$. Neither is this possible for sup $\mathbf{c} = \{i_b, j_b\}$. In case sup $\mathbf{c} = \{i_2, j_4\}$, the only possibility that $\mathbf{c} + \mathbf{z}' + \mathbf{z}'' = \mathbf{0}$ is that $\mathbf{z}' = \mathbf{e}_0 + \mathbf{e}_{k_2} + \mathbf{e}_0 = \mathbf{e}_{k_2}$ with $i = k$, and $\mathbf{z}'' = \mathbf{e}_{j_4}$. It then follows from the general form of the basis vectors that $\mathbf{c}$ is the sum of an odd number of basis vectors. This number is greater than 3, since otherwise $i = j$ and $\mathcal{T}' = \varnothing$ which is excluded. Since

$$k_4 = (k + 1)_4 = \ldots = (k + a)_4,$$

we also have $j < i$. So, $\mathbf{c} = \mathbf{b}_{j-1} + \sum_{l \geq j} \mathbf{b}_l$, according to Theorem 2.5 (cf. also (4.13)).

Because of the condition of our Theorem, at least one of the integers $(j - 1)_2$, $(j - 1)_3$ and $(j - 1)_4$ does not occur in blocks B_l, $l \geq j$, which leads to a contradiction. So, $\mathbf{c}$ is of weight 4 or 6 and the assumption $\mathbf{c} + \mathbf{z}' + \mathbf{z}'' = \mathbf{0}$ leaves the following possibilities (cf. also the proof of Theorem 4.2):

(*a*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*b*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*c*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_{j_4}$;

(*d*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{i_2} + \mathbf{e}_{i_3} + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*e*) $\mathbf{c} = \mathbf{e}_{k_2} + \mathbf{e}_{j_2} + \mathbf{e}_{j_3} + \mathbf{e}_{j_4}$;

(*f*) $\mathbf{c} = \mathbf{e}_{i_1} + \mathbf{e}_{k_2} + \mathbf{e}_0 + \mathbf{e}_{k_4}$;

(*g*) $\mathbf{c} = \mathbf{e}_{i_2} + \mathbf{e}_{k_2} + \mathbf{e}_{i_3} + \mathbf{e}_{k_4}$.

Because of the general form of the basis vectors, we conclude that codewords of types (*e*), (*f*) and (*g*) do not exist. The cases (*a*), (*c*) and (*d*) can be reduced to the weight-2 case, by considering $\mathbf{c} + \mathbf{b}_j$, $\mathbf{c} + \mathbf{b}_i$ and $\mathbf{c} + \mathbf{b}_i + \mathbf{b}_j$, respectively, and which can be eliminated as before.

Finally, if $\mathbf{c}$ is of type (*b*), then $\mathbf{c}$ must be the sum of an odd number of basis vectors, and so $i = j$ or $i > 1, j > 1$ or both (cf. Theorem 2.5). Now $i = j$ yields $\mathbf{c} = \mathbf{b}_i$, contradicting Theorem 2.5(*iii*). Let w.l.o.g. $1 < i < j$. Then we can write $\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \in X} \mathbf{b}_l$, where X is some subset of $\{i, i + 1, , \ldots, k + a\}$ of even size. Again, it follows

from the condition of the Theorem that B_{i-1} contains at least one integer which is not present in any of the blocks B_l, $l \geq i$. If $(i-1)_1$ does not occur anymore, it is contained in sup **c** and hence $i_1 = (i-1)_1$, which is a contradiction since $i > i-1$. Similar contradictions arise if one of $(i-1)_2$, $(i-1)_3$ or $(i-1)_4$ does not occur anymore in blocks B_l, $l \geq i$. We conclude that the assumption $d(\mathbf{x}, \mathbf{y}) = 0$ is false and hence, $\mathcal{B}^{ext}$ generates a Gray code with $|\overline{S}_{k+a}(\mathcal{B}^{ext})| = 2^{k+a+2} = 2^n$ words, i.e. a complete Gray code.

□

The following theorem applies Theorem 2.6 to the standard Gray code $G(k+a)$ with i replaced by k. The a weight-2 vectors $\mathbf{e}_{k,k+1}, \mathbf{e}_{k,k+2}, \ldots, \mathbf{e}_{k,k+a}$ of $G(k+a)$ correspond to a independent vectors $\mathbf{s}_1 = \mathbf{b}_k + \mathbf{b}_{k+1}$, $\mathbf{s}_2 = \mathbf{b}_k + \mathbf{b}_{k+2}$, …, $\mathbf{s}_a = \mathbf{b}_k + \mathbf{b}_{k+a}$, respectively, of the code $\mathcal{C}^{ext}$. These a vectors constitute a basis for a linear $[n, a, 2]$-code, which is an invariance group $\mathcal{G} = <\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_a>$ of the cover of Q_n obtained by translating the snake $\mathcal{S}$ of length 2^{k+2} over the 2^a vectors of $\mathcal{G}$. The 2^a sublists $G_k^j(k+a)$, $j = 0, 1, \ldots, 2^a - 1$, of $G(k+a)$ correspond to the 2^a open snakes expressed by (8.12).

Theorem 8.2

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ and $\mathcal{B}^{ext} = (B_1, B_2, \ldots, B_k, B_{k+1}, \ldots, B_{k+a})$ be two block lists that satisfy the conditions of Theorem 8.1 *and let $\mathcal{B}$ generate a snake of size 2^{k+2}. Let furthermore $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{2^a-1}$ be the 2^a words in the word list $\mathcal{S}^{ext}$ generated by $\mathcal{B}^{ext}$ at the positions $2^{k+2}-1, 2\cdot 2^{k+2}-1, 3\cdot 2^{k+2}-1, \ldots, 2^a\cdot 2^{k+2}-1$. Then one obtains a 2^a-cover by symmetric snakes of Q_n by carrying out the cyclic permutation $(\mathbf{u}_0\ \mathbf{u}_1\ \ldots\ \mathbf{u}_{2^a-1})$ in $\mathcal{S}^{ext}$. This cover has an invariance translation group $\mathcal{G}$ of size 2^a generated by the vectors*

$$\mathbf{b}_k + \mathbf{b}_{k+1}, \quad \mathbf{b}_k + \mathbf{b}_{k+2}, \quad \ldots, \quad \mathbf{b}_k + \mathbf{b}_{k+a}.$$

Proof. By applying Theorem 2.6 with necessary adaptations, e.g. by replacing the integer $k + t_j$, $j = 1, 2, \ldots, a$, in the transition sequence $\overline{S}_{k+a}$ by the ordered sequence of four integers of the corresponding block B_{k+t_j}, we partition $\overline{S}_{k+a}(\mathcal{B}^{ext})$, starting from its first element into 2^a subsequences of 2^{k+2} integers.

From the properties of the standard Gray code $G(k+a)$ and from the specific form of the blocks $B_k, B_{k+1}, \ldots, B_{k+a}$, which differ only in their third elements, these 2^a subsequences are all equal to $\overline{S}_k(\mathcal{B})$, apart from their last but one element. As a result, these subsequences are the transition sequences of 2^a open snakes such that each open snake, apart from their last words $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{2^a-1}$, can be obtained from any other one by a translation over some fixed

vector. These translation vectors are the 2^a linear combinations of the vectors $\mathbf{s}_1 = \mathbf{b}_k + \mathbf{b}_{k+1}$, $\mathbf{s}_2 = \mathbf{b}_k + \mathbf{b}_{k+2}$, …, $\mathbf{s}_a = \mathbf{b}_k + \mathbf{b}_{k+a}$,

More precisely, let

$$\mathcal{S}^0, \mathbf{u}_0, \quad \mathcal{S}^1, \mathbf{u}_1, \quad \ldots, \quad \mathcal{S}^{2^a-1}, \mathbf{u}_{2^a-1} \tag{8.12}$$

be the above mentioned open snakes. It follows rather easily from the properties of the Gray code $G(k + a)$ that (cf.(2.15))

$$\mathcal{S}^i = \mathcal{S}^{i-1} + \mathbf{s}_{t_i}, \qquad 1 \le i \le 2^a, \tag{8.13}$$

where the indices t_i are the elements of the complete transition sequence of the standard Gray code $G(a)$

$$\overline{S}_a = t_1, t_2, t_3, \ldots, t_{2^a}, \tag{8.14}$$

the concrete form of which is (cf. (2.17)).

$$\overline{S}_a = 1, 2, 1, \ldots, a, 1, 2, 1, \ldots, a \tag{8.15}$$

In (8.13), we identify $\mathcal{S}^{2^a}$ and $\mathcal{S}^0$. A similar rule

$$\mathbf{u}_i = \mathbf{u}_{i-1} + \mathbf{s}_{t_{i+1}}, \qquad 1 \le i \le 2^a. \tag{8.16}$$

holds for the last words $\mathbf{u}_i$ and $\mathbf{u}_{i-1}$ of the respective open snakes $\mathcal{S}^i, \mathbf{u}_i$, and $\mathcal{S}^{i-1}, \mathbf{u}_{i-1}$. Furthermore, the Hamming distance between $\mathbf{u}_{i-1}$ and the first word of $\mathcal{S}^{i+1}$ is equal to 1 for all relevant values of i, since these words are successive words in the Gray code generated by $\overline{S}_{k+a}(\mathcal{B}^{ext})$, according to Theorem 8.1. The Hamming distance between $\mathbf{u}_i$ and the last word of $\mathcal{S}^{i+1}$ is equal to

$$\| \mathbf{s}_{t_{i+1}} + \mathbf{e}_{(k+t_{i+1})_3} \| = \| \mathbf{b}_k + \mathbf{b}_{k+t_{i+1}} + \mathbf{e}_{(k+t_{i+1})_3} \| = \| \mathbf{e}_{k_3} \| = 1.$$

So, if we permute $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{2^a-1}$ in cyclic sense over one position to the right, we obtain the concatenation of 2^a closed snakes

$$\mathcal{S}^0, \mathbf{u}_{2^a-1}, \quad \mathcal{S}^1, \mathbf{u}_0, \quad \mathcal{S}^2, \mathbf{u}_1, \quad \ldots, \mathcal{S}^{2^a-1}, \mathbf{u}_{2^a-2}, \tag{8.17}$$

and these snakes are identical to

$$\mathcal{S}, \quad \mathcal{S} + \mathbf{s}_1, \quad \mathcal{S} + \mathbf{s}_1 + \mathbf{s}_2, \ldots, \mathcal{S} + \textstyle\sum_{t_i \in X_i} \mathbf{s}_{t_i}, \ldots, \mathcal{S} + \mathbf{s}_a,$$

where X_i is the multiset of the first i integers of (8.14) and where the (list) snake $\mathcal{S}$ is generated by $\overline{S}_k(\mathcal{B})$. This proves the Theorem. □

Example 8.3

Let $\mathcal{B}$ be the block list presented in Fig. 7.3 and let

$$\mathcal{B}^{ext} = (\mathcal{B}, B_{12}, B_{13}, B_{14}),$$

be its extension with the three additional blocks

$$B_{12} = (0, 6, 0, 13), \quad B_{13} = (0, 6, 8, 13), \quad B_{14} = (0, 6, 9, 13).$$

Since $\mathcal{B}$ generates a snake (Theorem 7.6) and since $\mathcal{B}^{ext}$ satisfies the condition of Theorem 8.1, we are entitled to apply Theorema 8.2. Hence, after having carried out the permutation

($\mathbf{u}_0$ $\mathbf{u}_1$ $\mathbf{u}_2$ … $\mathbf{u}_7$) of the words which are at the positions $2^{13} - 1$, $2 \cdot 2^{13} - 1$, $3 \cdot 2^{13} - 1$, …, $8 \cdot 2^{13} - 1$, respectively, we obtain a concatenation of eight (closed) snakes which together constitute a cover of Q_{16}. This cover has an invariance group

$$\begin{aligned} \mathcal{G} &= <\mathbf{b}_{11} + \mathbf{b}_{12},\ \mathbf{b}_{11} + \mathbf{b}_{13},\ \mathbf{b}_{11} + \mathbf{b}_{14}> \\ &= <\mathbf{e}_0 + \mathbf{e}_{11},\ \mathbf{e}_8 + \mathbf{e}_{11},\ \mathbf{e}_9 + \mathbf{e}_{11}> \\ &= <\mathbf{e}_0 + \mathbf{e}_8,\ \mathbf{e}_0 + \mathbf{e}_9,\ \mathbf{e}_0 + \mathbf{e}_{11}>. \end{aligned}$$

This is precisely the 8-cover of Theorem 7.7.

We emphasize that extending $\mathcal{B}$ with the blocks

$$B_{12} = (0, 6, 0, 13), \quad B_{13} = (0, 6, 8, 13), \quad B'_{14} = (0, 6, 10, 13).$$

will not provide us with a cover. This is because the extended block list no longer satisfies the conditions of Theorem 8.1. In particular, block B_7 does not contain an integer that does not occur in blocks B_l, $l > 7$. In Section 7.2, we already established, by applying different tools, that the corresponding family of snakes (7.23) is not a cover (cf. also Remark 7.1).

We remark that the mutual order of the blocks B_{k+1}, B_{k+2}, …, B_{k+a} is not essential. The proofs of Theorem 8.1 and Theorem 8.2 do not depend on their order. If some block list $\mathcal{B}^{ext}$ satisfies the conditions of Theorem 8.2, we shall say that $\mathcal{B}^{ext}$ *generates* a symmetric 2^a-cover of Q_n.

Theorem 8.3

Let $\mathcal{B} = (B_1, B_2, \ldots, B_k)$ be a block list that satisfies the conditions of Theorem 5.8 *and let k_0 be the number of blocks B_j which only satisfy requirement* (*iii*) *of that Theorem and not* (*i*) *and* (*ii*). *If*

$$k_0 \le k + |I_3| + 2 - n,$$

then $\mathcal{B}$ (*or an equivalent list*) *can be extended to a list $\mathcal{B}^{ext}$ which generates a 2^a-cover of Q_n with $a = n - k - 2$.*

Proof. From Theorem 5.8 we know that $\mathcal{B}$ (or an equivalent list) generates a snake in Q_n. From Theorem 8.1 and Theorem 8.2 it follows that there exists an extended list $\mathcal{B}^{ext}$ generating a 2^a-cover, if one can find $a - 1 = n - k - 3$ integers $l_3 \in I_3$ that are not in one of the k_0 blocks as described in the Theorem. Since B_k is not one of these k_0 blocks, we must have

$$|I_3| - 1 - k_0 \ge n - k - 3. \qquad \square$$

Example 8.4

The list $\mathcal{B}$ in Fig. 7.3 satisfies the conditions of Theorem 5.8. Block B_7 is the only block which satisfies requirement (*iii*) of that Theorem and not (*i*) or (*ii*). So, $k_0 = 1$. Since $1 \le 11 + 4 + 2 - 16$, $\mathcal{B}$ can be transformed and extended to a list that generates a 2^3-cover of Q_{16}. Notice

that Theorem 8.3 cannot be applied to a list like the one of Fig. 5.13 since the conditions of Theorem 5.8 do not hold.

To summarize this section, we emphasize once more that the vectors $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k$ that correspond to a list $\mathcal{B}$ generate a code $\mathcal{C}$ with minimum distance 4. This minimum distance garantees (cf. Theorem 4.2) the separability condition for a snake. Extending $\mathcal{C}$ to $\mathcal{C}^{ext}$ by adding basis vectors $\mathbf{b}_{k+1}, \ldots, \mathbf{b}_{k+a}$ decreases the minimum distance from 4 to 2. Consequently, the separability condition in the generated transition sequence $\overline{S}_{k+a}(\mathcal{B}^{ext})$ is no longer valid, though the nearness condition still holds. So, $\overline{S}_{k+a}(\mathcal{B}^{ext})$ still generates a Gray code, and this Gray code can be cut into 2^a disjoint pieces such that each piece is a snake, due to the fact that these pieces correspond to the subcode $\mathcal{C}$ of $\mathcal{C}^{ext}$.

8.3 Special Bases for $R(m-2, m)$ and Covers of Q_n

In this section, we shall prove that the linear code $R(m-2, m)$ (the Extended Hamming code $\mathcal{H}_m^{ext}$) always has a minimum-weight basis (of weight 4) such that it satisfies the fixed-position property w.r.t. some given parallel system in EG(m, 2), for all $m > 2$. We know already, by construction, that such a basis exists for $m = 3$ and $m = 4$ (cf. Chapter 5).

First we repeat, for our convenience, some definitions and conventions introduced in [35] and applied in Chapter 5. Let the dimension of $R(m-2, m)$ be equal to k. So, this code contains 2^k codewords. The length of these words is equal to $n = 2^m$, which is the number of points in EG(m, 2). These points are labeled from 0 until $2^m - 1$, so the point set of EG(m, 2) is written as

$$\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{2^m-1}\}.$$

or as

$$\{P_0, P_1, \ldots, P_{2^m-1}\},$$

(cf. [35, Section 3.2]).

In Section 5.2, we adopted the convention that P_i stands for the reversed binary representation of i, for $0 \le i \le 2^m - 1$. Since a codeword $\mathbf{v} \in R(m-2, m)$ is by definition the characteristic vector of a subset of these points, we can identify $\mathbf{v}$ with the index set sup $\mathbf{v}$.

Let furthermore $\mathcal{P} = \{I_1, I_2, I_3, I_4\}$ be a parallel system of $(m-2)$-flats that covers EG(m, 2). Here, we represent the point set I_i, $i \in \{1, 2, 3, 4\}$, also by the index set of the points it contains. We assume that I_1 stands for a linear subspace of EG(m, 2) of dimension $m-2$, i.e. it is a subspace containing $\mathbf{p}_{k_1} := \mathbf{0}$, whereas for some $k_i \in \{0, 1, \ldots, 2^m - 1\}$ with $i \in \{2, 3, 4\}$, $I_i = I_1 + \mathbf{p}_{k_i}$ is one of its cosets.

Just like in the previous sections, we shall represent the index values 0, 1, …, $2^m - 1$ by their reversed binary representation, and so we write $I_i = I_1 \oplus k_i$, where $\oplus$ denotes *Nim addition*. We shall say that a coset S has a *zero-Nim sum* if S is a linear subspace of EG(m, 2), the sum of its vectors is **0**. Hence, the Nim sum of the labels of these vectors is 0. An immediate consequence is that the same is true for any coset.

Theorem 8.4

Let $\mathcal{P} = \{I_1, I_2, I_3, I_4\}$ *be some parallel system of* $(m-2)$*-flats covering* EG(m, 2) *and let* B *be a* 2*-flat. Then the intersections of* B *with the flats* I_i *satisfy one of the following relations:*

(i) $|B \cap I_i| = 1$, *for all* $i \in \{1, 2, 3, 4\}$;

(ii) $|B \cap I_i| = |B \cap I_j| = 2$, *for some* $i, j \in \{1, 2, 3, 4\}$ *and* $|B \cap I_k| = 0$, *for* $k \in \{1, 2, 3, 4\} \setminus \{i, j\}$;

(iii) $|B \cap I_i| = 4$, *for some* $i \in \{1, 2, 3, 4\}$ *and* $|B \cap I_j| = 0$, *for* $j \in \{1, 2, 3, 4\} \setminus \{i\}$.

Proof. Assume that there is an $i \in \{1, 2, 3, 4\}$ such that $B = \{a, b, c, d\}$ has at least two points - say a and b - in common with I_i. Then it follows that $a \oplus b \in I_1$ and hence, $c \oplus d \in I_1$, since $a \oplus b \oplus c \oplus d = 0$. But then a and d are both in j, for some $j \in \{1, 2, 3, 4\}$. If $i = j$, we are in case (*iii*), and if $i \neq j$ we are in case (*ii*). If our assumption is false, we are in case (*i*). □

Theorem 8.5

For the code $R(m-2, m)$ *one can always find a minimum-weight basis* $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_1, \ldots, \mathbf{b}_k)$ *such that the ordered blocks* B_i *corresponding to sup* $\mathbf{b}_i$, $1 \leq i \leq k$, *all satisfy the fixed-position property with respect to any parallel system* $\mathcal{P}$ *of* $(m-2)$*-flats covering* EG(m, 2).

Proof Let $\mathcal{P}$ be some given parallel system. Let B be a 2-flat which is of type (*ii*) (cf. Th. 8.4) with respect to $\mathcal{P}$, e.g. $B = (a_1, b_1, a_2, b_2)$, where $a_1, b_1 \in I_1$ and $a_2, b_2 \in I_2$ (remember that the a_i and b_i are binary numbers). We remark that any 2-flat is uniquely determined by three of its four points and that any of its points is the Nim sum of the others when using the binary representation.

Hence, for any $c_3 \in I_3$, the block (a_1, a_2, c_3, c_4) with $c_4 = a_1 \oplus a_2 \oplus c_3$ stands for a 2-flat. Since $a_1 \oplus a_2 \oplus b_1 \oplus b_2 = 0$, it follows that $b_1 \oplus b_2 \oplus c_3 \oplus c_4 = 0$ and hence (b_1, b_2, c_3, c_4) also represents a 2-flat, and $c_4 \in I_4$ by Theorem 8.4. Now, we can write $(a_1, b_1, a_2, b_2) = (a_1, a_2, c_3, c_4) \Delta (b_1, b_2, c_3, c_4)$, which shows that B can be written as the sum of two blocks satisfying the fixed-position property. The same holds for any other block which represents a 2-flat of type (*ii*).

Similar arguments can be raised for 2-flats of type (*iii*), say for (a_1, b_1, c_1, d_1). For any $e_2, s_2 \in I_2$ and any $f_3 \in I_3$, we can find a $t_3 \in I_3$ such that

$$(a_1, b_1, c_1, d_1) = (a_1, e_2, f_3, g_4) \Delta (b_1, e_2, f_3, h_4) \Delta (c_1, s_2, t_3, g_4) \Delta (d_1, s_2, t_3, h_4),$$

and where the four blocks on the RHS all represent 2-flats. This also illustrates that all blocks representing a 2-flat of type (*iii*) can be written as a sum of blocks satisfying the fixed-position property.

We know that $R(m-2, m)$ is spanned by its weight-4 vectors, i.e. by some of the 2-flats in EG(m, 2) (see e.g. [35, Theorem 3.2.3 and 3.2.4]). From the above considerations it now follows that $R(m-2, m)$ is also spanned by the subset of its weight-4 vectors the corresponding blocks of which satisfy the fixed-position property. □

Example 8.5

We shall construct a basis for the code $R(3, 5)$ which has dimension (cf. (5.5))

$$k = \sum_{i=0}^{3} \binom{5}{i} = 26.$$

For the parallel system $\mathcal{P} = \{I_1, I_2, I_3, I_4\}$, we take

$$I_1 = \{\ 0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7\},$$
$$I_2 = \{\ 8,\ 9, 10, 11, 12, 13, 14, 15\},$$
$$I_3 = \{16, 17, 18, 19, 20, 21, 22, 23\},$$
$$I_4 = \{24, 25, 26, 27, 28, 29, 30, 31\}.$$

It is obvious that the following (ordered) blocks correspond to independent vectors (we shall speak of *independent blocks* in the next).

$B_1 = (7,\ 8, 16, 31)$ $\qquad$ $B_{12} = (0, 12, 16, 28)$

$B_2 = (6,\ 8, 16, 30)$ $\qquad$ $B_{13} = (0, 13, 16, 29)$

$B_3 = (5,\ 8, 16, 29)$ $\qquad$ $B_{14} = (0, 14, 16, 30)$

$B_4 = (4,\ 8, 16, 28)$ $\qquad$ $B_{15} = (0, 15, 16, 31)$

$B_5 = (3,\ 8, 16, 27)$ $\qquad$ $B_{16} = (0, 15, 17, 30)$

$B_6 = (2,\ 8, 16, 26)$ $\qquad$ $B_{17} = (0, 15, 18, 29)$

$B_7 = (1,\ 8, 16, 25)$ $\qquad$ $B_{18} = (0, 15, 19, 28)$

$B_8 = (0,\ 8, 16, 24)$ $\qquad$ $B_{19} = (0, 15, 20, 27)$

$B_9 = (0,\ 9, 16, 25)$ $\qquad$ $B_{20} = (0, 15, 21, 26)$

$B_{10} = (0,10, 16, 26)$ $\qquad$ $B_{21} = (0, 15, 22, 25)$

$B_{11} = (0,11, 16, 27)$ $\qquad$ $B_{22} = (0, 15, 23, 24)$

In order to find additional independent blocks, we can take e.g. three more blocks containing 0 and 14:

$B_{23} = (0, 14, 22, 24),$ $\qquad$ $B_{24} = (0, 14, 20, 26),$ $\qquad$ $B_{25} = (0,14, 18, 28).$

Notice that the remaining blocks containing 0 and 14 depend on the blocks we already have:

$$(0, 14, 17, 31) = B_{14} \Delta B_{15} \Delta B_{16},$$
$$(0, 14, 19, 29) = B_{14} \Delta B_{17} \Delta B_{18},$$
$$(0, 14, 21, 27) = B_{14} \Delta B_{19} \Delta B_{20},$$
$$(0, 14, 23, 25) = B_{14} \Delta B_{21} \Delta B_{22}.$$

Finally, to complete the basis, we can take

$$B_{26} = (0, 13, 20, 25).$$

Notice that other blocks containing 0 and 13 can be obtained as sums of the blocks B_1, B_2, …, B_{26}. We can write e.g. $(0, 13, 18, 31) = B_{13} \Delta B_{15} \Delta B_{17}$ and similarly for other blocks of type

$$(0, 13, \ldots, \ldots).$$

The following theorem appears to be useful for the construction of minimum-weight bases for $R(m - 2, m)$ codes. Actually, this theorem is nothing else than Theorem 5.6 for $r = m - 2$ in terms of Nim sums (cf. also [23], Chapter 13).

Theorem 8.6

Let $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_1, \ldots, \mathbf{b}_k)$ *with* $k = \sum_{i=0}^{m-2} \binom{m}{i}$ *be a set of independent binary vectors of length* $n = 2^m$ *and of weight* 4. *If the corresponding blocks* B_i *have zero-Nim-sum then* **B** *generates the Reed-Muller code* $R(m - 2, m)$.

Proof. Let $\mathcal{C}$ be the $[n, k, d]$-code generated by **B**. Let $\mathbf{v} = \sum_j c_j \mathbf{b}_j$, $c_j \in \{0, 1\}$, be some word of $\mathcal{C}$, and let $V := \sup \mathbf{v}$. Since $V = \sum_j^{\Delta} c_j \sup \mathbf{b}_j$, where Σ^{Δ} stands for repeated symmetric difference, V also has zero-Nim-sum. Furthermore, since all $\mathbf{b}_i$ have even weight, the total number $|V|$ of integers in V is also even. Because the Nim sum of two different integers cannot be equal to 0, it follows that either $|V| = 0$ or $|V| \geq 4$, and hence $d = 4$. So, $\mathcal{C}$ is an $[n, k, 4]$-code with $k = \sum_{i=0}^{m-2} \binom{m}{i}$. Since the code is generated by its minimum-weight-4 basis vectors corresponding to 2-flats of EG$(m, 2)$, we may conclude that $\mathcal{C}$ is equivalent to $R(m - 2, m)$. □

Theorems 8.5 and 8.6, enable us to compose block lists satisfying the fixed-position property, and which correspond to an minimum-weight basis basis for $R(m - 2, m)$.

Example 8.6

One can verify that the block list in Example 8.5 satisfies the conditions of Theorem 8.6, and so the code it generates is indeed $R(3, 5)$. The following block list $\mathcal{B}$ also corresponds to $R(3,5)$ (This code, though equivalent, need not be identical to the former one).

$B_1 = (\underline{7},\ 8, 16, 31)$, $B_{12} = (0, 15, 16, \underline{31})$,
$B_2 = (\underline{6},\ 8, 16, 30)$, $B_{13} = (0, 14, \underline{16}, 30)$,
$B_3 = (\underline{5},\ 8, 16, 29)$, $B_{14} = (0, 15, 18, \underline{29})$,
$B_4 = (\underline{4},\ 8, 16, 28)$, $B_{15} = (0, 14, \underline{18}, 28)$,
$B_5 = (\underline{3},\ 8, 16, 27)$, $B_{16} = (0, 15, 20, \underline{27})$,
$B_6 = (\underline{2},\ 8, 16, 26)$, $B_{17} = (0, 14, \underline{20}, 26)$,
$B_7 = (\underline{1},\ 8, 16, 25)$, $B_{18} = (0, 15, 22, \underline{25})$,
$B_{19} = (0, \underline{14}, 22, 24)$,
$B_8 = (0,\ \underline{8}, 16, 24)$, $B_{20} = (0, 15, 17, \underline{30})$,
$B_9 = (0,\ \underline{9}, 16, 25)$, $B_{21} = (0, 13, \underline{17}, 28)$,
$B_{10} = (0, \underline{10}, 16, 26)$, $B_{22} = (0, 15, 21, \underline{26})$,
$B_{11} = (0, \underline{12}, 16, 28)$, $B_{23} = (0, \underline{13}, 21, 24)$,
$B_{24} = (0, 15, 19, \underline{28})$,
$B_{25} = (0, \underline{11}, 19, 24)$,
$B_{26} = (0, 15, 23, 24)$.

Each block B_i, $1 \le i \le 26$, contains at least one integer which does not occur anymore in blocks B_l, $l > i$. So, these integers can be considered as pivots, and are marked by an underscore. This implies that we have 26 independent blocks. Since all blocks have zero-Nim-sum, the corresponding basis $\mathcal{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{26})$ generates the Reed-Muller code $R(3, 5)$. Moreover, the list satisfies the conditions of Theorem 5.8, and so it either ifself generates a snake of length 2^{28} in Q_{32} or it is equivalent to such a list.

The blocks which satisfy requirement (*iii*) of Theorem 5.8, and not (*i*) or (*ii*), are B_{13}, B_{15}, B_{17} and B_{21}. So, the number k_0 of Theorem 8.3 is equal to 4. Since $k + |I_3| + 2 - n = 26 + 8 + 2 - 32 = 4$, we can extend $\mathcal{B}$ to $\mathcal{B}^{ext}$ with four additional blocks such that $\mathcal{B}^{ext}$ generates a symmetric 16-cover of Q_{32}, according to Theorem 8.3. The additional blocks are

$B_{27} = (0, 15,\ 0, 24)$,
$B_{28} = (0, 15, 22, 24)$,
$B_{29} = (0, 15, 21, 24)$,
$B_{30} = (0, 15, 19, 24)$.

Example 8.7

We present once more a block list $\mathcal{B}$ for the code $R(2, 4)$. This one is quite similar to the list for $R(3, 5)$ in Example 8.6, and satisfies the conditions of Theorem 5.8.

$B_1 = (\underline{3}, 4,\ 8, 15)$, $B_5 = (0, 7,\ 8, \underline{15})$,
$B_2 = (\underline{2}, 4,\ 8, 14)$, $B_6 = (0, 6,\ \underline{8}, 14)$,
$B_3 = (\underline{1}, 4,\ 8, 13)$, $B_7 = (0, 7,10, \underline{13})$,
$B_8 = (0, \underline{6},10, 12)$,
$B_4 = (0, \underline{4},\ 8, 12)$, $B_9 = (0, 7,\ 9, \underline{14})$,
$B_{10} = (0, \underline{5},\ 9, 12)$,
$B_{11} = (0, 7,11, 12)$.

According to Theorem 8.3 this list can be extended with blocks

$B_{12} = (0, 7, 0, 12)$, $B_{13} = (0, 7, 9, 12)$, $B_{14} = (0, 7, 10, 12)$

and the extended list $\mathcal{B}^{ext}$ generates a symmetric Gray code which is a concatenation of eight open snakes. From Theorem 8.2 we know that we can interchange the last words of these

snakes to obtain an 8-cover consisting of eight closed (or cyclic) snakes. According to our remark right after Example 8.3, we say that $\mathcal{B}^{ext}$ generates a symmetric 8-cover.

We are now ready to formulate and to prove a general theorem for the existence of symmetric covers of Q_n.

Theorem 8.7

For the Reed-Muller code $R(m-2, m)$ of word length n $(= 2^m)$ and of dimension k $(= 2^m - m - 1)$, $m \geq 3$, one can always construct a weight-4 basis satisfying the fixed-position property such that the corresponding block list $\mathcal{B}$ generates a snake of length 2^{k+2} in Q_n, and which can be extended to a list $\mathcal{B}^{ext}$ generating a symmetric 2^{m-1}-cover of Q_n.

Proof. For the four subsets I_1, I_2, I_3, I_4 (cf. Definiton 4.1) of $\{0, 1, \ldots, m-1\}$, we take

$$\begin{aligned}
I_1 &= \{\ 0, && 1, && \ldots, && 2^{m-2}-1 \ \}, \\
I_2 &= \{\ 2^{m-2}, && 2^{m-2}+1 && , && \ldots, && 2^{m-1}-1 \ \}, \\
I_3 &= \{\ 2^{m-1}, && 2^{m-1}+1 && , && \ldots, && 2^{m-1}+2^{m-2}-1\}, \\
I_4 &= \{2^{m-1}+2^{m-2}, && 2^{m-1}+2^{m-2}+1, && \ldots, && 2^m-1 \ \}.
\end{aligned}$$

The list $\mathcal{B}$ to be constructed will consist of three sublists $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$. Sublist $\mathcal{B}_1$ is defined as

$$\mathcal{B}_1 = \begin{cases}
(\underline{2^{m-2}-1}, & 2^{m-2} & , & 2^{m-1} & , & 2^m-1 &), \\
(\underline{2^{m-2}-2}, & 2^{m-2} & , & 2^{m-1} & , & 2^m-2 &), \\
\qquad\vdots & & \vdots & & \vdots & & \\
(\ \underline{1}, & 2^{m-2} & , & 2^{m-1} & , & 2^{m-1}+2^{m-2}+1 &).
\end{cases}$$

The sublist $\mathcal{B}_3$ is built up as follows

$$\mathcal{B}_3^1 = \begin{cases}
(\ 0, & 2^{m-1}-1 & , & 2^{m-1} & , & \underline{2^m-1} &), \\
(\ 0, & 2^{m-1}-2 & , & \underline{2^{m-1}} & , & 2^m-2 &), \\
(\ 0, & 2^{m-1}-1 & , & 2^{m-1}+2 & , & \underline{2^m-3} &), \\
(\ 0, & 2^{m-1}-2 & , & \underline{2^{m-1}+2} & , & 2^m-4 &), \\
\qquad\vdots & & \vdots & & \vdots & & \\
(\ 0, & 2^{m-1}-1 & , & 2^{m-1}+2^{m-2}-2 & , & \underline{2^m-2^{m-2}+1} &), \\
(\ 0, & \underline{2^{m-1}-2} & , & 2^{m-1}+2^{m-2}-2 & , & 2^m-2^{m-2} &),
\end{cases}$$

$$\mathcal{B}_3^2 = \begin{cases} (\ 0,\ 2^{m-1}-1\ ,\ 2^{m-1}+1\ ,\ \underline{2^m-2}\ ), \\ (\ 0,\ 2^{m-1}-3\ ,\ \underline{2^{m-1}+1}\ ,\ 2^m-4\ ), \\ (\ 0,\ 2^{m-1}-1\ ,\ 2^{m-1}+5\ ,\ \underline{2^m-6}\ ), \\ (\ 0,\ 2^{m-1}-3\ ,\ \underline{2^{m-1}+5}\ ,\ 2^m-8\ ), \\ \quad\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ (\ 0,\ 2^{m-1}-1\ ,\ 2^{m-1}+2^{m-2}-3\ ,\ \underline{2^m-2^{m-2}+2}\ ), \\ (\ 0,\ \underline{2^{m-1}-3}\ ,\ 2^{m-1}+2^{m-2}-3\ ,\ 2^m-2^{m-2}\ ), \end{cases}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$\mathcal{B}_3^{m-2} = \begin{cases} (\ 0,\ 2^{m-1}-1\ ,\ 2^{m-1}+2^{m-3}-1\ ,\ \underline{2^m-2^{m-3}}\ ), \\ (\ 0,\ \underline{2^{m-1}-2^{m-3}-1},\ 2^{m-1}+2^{m-3}-1\ ,\ 2^m-2^{m-2}\ ), \end{cases}$$

$$\mathcal{B}_3^{m-1} = (\ 0,\ 2^{m-1}-1\ ,\ 2^{m-1}+2^{m-2}-1\ ,\ 2^m-2^{m-2}\ ).$$

As one can verify, the sublists $\mathcal{B}_3{}^i$ have size 2^{m-i-1}, $1 \le i \le m-1$. For each block, we mark the integer which plays the role of pivot by an underscore. These pivots occur alternately in the third and in the fourth column, except when we are dealing with the last block in $\mathcal{B}_3{}^i$, where $1 \le i < m-1$. For those blocks we selected the integer in the second column. The total number of blocks in $\mathcal{B}_3$ is

$$\sum_{i=1}^{m-1} 2^{m-i-1} = 2^{m-1} - 1.$$

Finally, sublist $\mathcal{B}_2$ consists of the blocks

$$(\ 0,\ \underline{i_2}\ ,\ 2^{m-1}\ ,\ 2^{m-1}-i_2\ ).$$

Here, i_2 runs through the set $I_2 \setminus I'_2$, where I'_2 is the subset of I_2 consisting of those integers that are already present in the blocks of $\mathcal{B}_3$. All integers in $I_2 \setminus I'_2$ are chosen to be pivots. The total number of blocks in $\mathcal{B}_2$ is equal to $|I_2 \setminus I'_2| = 2^{m-2} - m + 1$. Hence, the block list $\mathcal{B} = \mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ contains

$$2^{m-2} - 1 + 2^{m-2} - m + 1 + 2^{m-1} - 1 = 2^m - m - 1$$

blocks. Since all these blocks have a different pivot, they are independent and since the Nim sum of the integers in each block is zero, they correspond to a minimum-weight basis of $R(m-2, m)$ (cf. Theorem 8.6). As one can easily verify they also satisfy the conditions of Theorem 5.8, and so $\mathcal{B}$ generates a snake of length 2^{k+2} in Q_n. Moreover, if we take the I_3-integers $i_3{}^1$, $i_3{}^2$, …, $i_3{}^{m-2}$, which are in the last blocks of $\mathcal{B}_3{}^1$, $\mathcal{B}_3{}^2$, …, $\mathcal{B}_3{}^{m-2}$, we can extend $\mathcal{B}$ with blocks

$$\begin{aligned}
\mathcal{B}_{k+1} &= (0,\ 2^{m-1}-1\ ,\ 0\ ,\ 2^m-2^{m-2}),\\
\mathcal{B}_{k+2} &= (0,\ 2^{m-1}-1\ ,\ i_3^1\ ,\ 2^m-2^{m-2}),\\
\mathcal{B}_{k+3} &= (0,\ 2^{m-1}-1\ ,\ i_3^2\ ,\ 2^m-2^{m-2}),\\
&\bullet\bullet\bullet\\
\mathcal{B}_{k+m-1} &= (0,\ 2^{m-1}-1\ ,\ i_3^{m-2}\ ,\ 2^m-2^{m-2}).
\end{aligned}$$

to a block list $\mathcal{B}^{ext}$.

Now according to Theorem 8.3, this list $\mathcal{B}^{ext}$ generates a symmetric 2^{m-1}-cover of Q_n or is equivalent to such a list. If $\mathcal{B}$ does not generate a snake immediately, but first has to be altered according to the procedure described in the proof of Theorem 5.8, there is a block $B_{i-1} = (0, (i-1)_2, (i-1)_3, (i-1)_4)$ and a codeword

$$\mathbf{c} = \mathbf{b}_{i-1} + \sum_{l \geq i} \mathbf{b}_l$$

with $\sup \mathbf{c} = \{0, i_2, p_3, j_4\}$, $i < j \leq k$. Since $(i-1)_3$ is a pivot in this case, it follows that $p_3 = (i-1)_3$. But then all integers l_3 in $\sum_{l \geq i}^{\oplus} B_l$ corresponding to $\sum_{l \geq i} \mathbf{b}_l$ have to occur an even number of times.

Because of the structure of the list (cf. Examples 8.6 and 8.7, and also Appendix D), the blocks B_l in the above sum occur as pairs of consecutive blocks. But each such pair contains an integer l_4, which is a pivot, so there can be only one such pair. However, a word $\mathbf{c} = \mathbf{b}_{i-1} + \mathbf{b}_j + \mathbf{b}_{j+1}$, $j \geq i$, always contains three integers of I_2, or three integers of I_4 (or both), and hence, we have a contradiction. Therefore, the list $\mathcal{B}$ itself generates a snake and need not be transformed into an equivalent list, which proves the Theorem. □

As for an illustration of Theorem 8.7 we once again refer to Example 8.7 with $m = 4$, to Example 8.6 with $m = 5$ and to Appendix D with $m = 6$.

Corollary 8.8

For any n which satisfies $2^{m-1} < n \leq 2^m$, one can construct a symmetric 2^{m-1}-cover of Q_n which is invariant under a translation group of order 2^{m-1}, for $m \geq 3$.

Proof. For $n = 2^m$, the result follows immediately by applying Theorem 5.8 and 8.2 to the block lists $\mathcal{B}$ and $\mathcal{B}^{ext}$ defined in the proof of Theorem 8.7. For $n = 2^m - y$, $0 < y < 2^{m-1}$, we omit the first y blocks from the list $\mathcal{B}$. The remaining list still satisfies the conditions of Theorems 5.8 and 8.2.

An alternative related method is the process of puncturing with respect to the underscored integers in blocks $B_1, B_2, \ldots, B_{2^{m-1}-1}$, respectively (cf. Section 7.1). □

We would like to point out that for $4 < n \leq 8$ and for $8 < n \leq 16$, Corrollary 8.8 gives better results than [32], i.e. the number of snakes in the cover of Q_n is equal to 4 and to 8, respectively, whereas in [32], it is only stated that this number is upperbounded by 16.

Even in the range $16 < n \leq 32$, one could say that Corollary 8.8 provides us with slightly better results, since the 16 covering snakes are symmetric, and so we have a *symmetric* 16-cover. Moreover, in the range $4 < n \leq 16$, it gives a supplement to a result in [2] which states that for any even integer $r \geq 4$, $n \geq 2$, the graph K_r^n, being the n^{th} power of the complete graph K_r, can be covered with r^3 snakes, be it that these snakes are not all mutually disjoint.

Appendices

Appendix A.

Upper and Lower Bounds For the Length of Snakes

Let $s(n)$ be the maximal length of a snake in Q_n. At present, the exact value of $s(n)$ has been determined only for six values of $s(n)$, i.e. $s(2) = 4$, $s(3) = 6$, $s(4) = 8$, $s(5) = 14$, $s(6) = 26$ and $s(7) = 48$.

In the course of time, several bounds have been derived. We mention the following review of several upper bounds found since 1987. In [31], Solov'jeva derived

$$s(n) \le \left(1 - \frac{2}{n^2 - n + 2}\right) 2^{n-1}, \qquad n \ge 7 \qquad \text{(A.1)}$$

Her method is based on counting four-cycles intersecting a fixed snake $\mathcal{S}$ in i vertices, $0 \le i \le 4$. In [30] Snevily improved this bound by deriving

$$s(n) \le \left(1 - \frac{2}{20n - 41}\right) 2^{n-1}, \qquad n \ge 12 \qquad \text{(A.2)}$$

His method is based on estimating the average number of vertices in $\mathcal{S}$ adjacent to a vertex not in $\mathcal{S}$.

By refining some details of the proof in [30], Emelyanov [7] found

$$s(n) \le \left(1 - \frac{1}{6n - 13}\right) 2^{n-1}, \qquad n \ge 19 \qquad \text{(A.3)}$$

By using some parts of Snevily's proof and Solov'jeva method, Emelyanov and Lukito in [8] showed that for $n \ge 7$,

$$s(n) \le \left(1 - \frac{2(n^5 - 18n^4 + 68n^3 - 7n^2 - 298n + 270)}{5n^6 - 31n^5 + 47n^4 - 57n^3 + 362n^2 - 806n + 540}\right) 2^{n-1}, \qquad \text{(A.4)}$$

$$\le \left(1 - \frac{2}{5n - 59 + o(1)}\right) 2^{n-1}.$$

For all n, Zemor in [42] proved that

$$s(n) \le 1 + \frac{6n}{6n + \frac{\sqrt{n}}{6\sqrt{6}} - 7} 2^{n-1} \qquad \text{(A.5)}$$

by applying a technique counting vertices of 'high degree' ouside $\mathcal{S}$. We remark that Zemor has introduced an essentially a new element, since his upper bound is not of the type $r(n)2^{n-1}$, where $r(n)$ is some rational function of n, but rather shows a $\sqrt{n}$ -dependence.

Although asymptotically Zemor's bound is the best bound so far, bound (A.4) is better on the interval $11 \le n \le 15607$. Moreover, Lukito and van Zanten in [20] proved

$$s(n) \leq \left(1 - \frac{2}{5n + 47 + \frac{450}{n-10}}\right) 2^{n-1} + \frac{16n - 80}{5n^2 - 3n - 20}, \qquad \text{(A.6)}$$

which is an improvement of bound (A.4) for $n \geq 7$ and which is better than (A.4) for $7 \leq n \leq 19081$.

For n-values with $8 \leq n \leq 20$, the following table of upper bounds is copied from [19, Table 2, p. 8]. Notice that for a particular n-value, any of the expressions A.1 - A.6 for upper bounds may not be the best one. In fact, for n-values with $n \geq 11$, the upper bounds listed in the table are lower than any upper bound given by any of the six expressions A.1 - A.6.

Word length (n)	Upper bound of $s(n)$	Word length (n)	Upper bound of $s(n)$
7	48*	14	8017
8	123	15	16063
9	249	16	32172
10	500	17	64425
11	994	18	128990
12	1995	19	258230
13	4000	20	516900

Tabel A.1

Lower bounds for $s(n)$ are mostly of the type $\lambda 2^n$. A bound of this type was first established by Evdovkimov in [10] with $\lambda = s(8)/2^9$. In [32], Wojciechowski obtained $\lambda = 9/64 = 0.1406\ldots$. A better lower bound is the one established by Abbot and Katchalski. In [1] they derived $s(n) \geq \lambda 2^n$ with $\lambda = 3.00781$.

With respect to general methods to construct maximal snakes, the most noticable lower bounds for $s(n)$ were produced by the research group from Artificial Intelligence Center at the Georgia University (http://ai.uga.edu/) where genetic algorithms have been developed for constructing maximal snakes. Kochut [17] and Potter [26] have successfully constructed the current maximal snakes in Q_n for $n = 7, 9, 10$ and 11 (cf. also [5]).

For $n = 8$, Paterson and Tuliani in [24] using the necklace approach, were the first authors who attained the current lower bound $96 \leq s(8)$. The most recent lower bounds are due to Cassella and Potter in [4, 5] who obtained new lower bounds $180 \leq s(9)$, $344 \leq s(10)$ and $630 \leq s(11)$, using a genetic algorithm, as mentioned earlier.

On the following page, we present a list of the best lower bounds for snakes known at this moment for $7 \leq n \leq 20$.

Word length (n)	Lower bound of $s(n)$	Word length (n)	Lower bound of $s(n)$
7	48*	14	4934
8	96	15	9868
9	180	16	19740
10	344	17	39480
11	630	18	78960
12	1238	19	157900
13	2468	20	315800

Tabel A.2

Appendix B

Some Covers For Small Hypercubes

Based on the block list $\mathcal{B} = (B_1, B_3, B_3, B_4)$ of Fig. 7.1, where

$$B_1 = (1, 3, 5, 7), \quad B_2 = (0, 2, 4, 6), \quad B_3 = (0, 3, 5, 6), \quad B_4 = (0, 3, 4, 7),$$

we constructed the following symmetric 4-cover of Q_8.

$\mathcal{S}^{(8)}$	$\mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_4$	$\mathcal{S}^{(8)} + \mathbf{e}_0 + \mathbf{e}_5$	$\mathcal{S}^{(8)} + \mathbf{e}_4 + \mathbf{e}_5$,
0. 00000000;	10001000;	10000100;	00001100; (1);
1. 01000000;	11001000;	11000100;	01001100; (3);
2. 01010000;	11011000;	11010100;	01011100; (5);
3. 01010100;	11011100;	11010000;	01011000; (7);
4. 01010101;	11011101;	11010001;	01011001; (0);
5. 11010101;	01011101;	01010001;	11011001; (2);
6. 11110101;	01111101;	01110001;	11111001; (4);
7. 11111101;	01110101;	01111001;	11110001; (6);
8. 11111111;	01110111;	01111011;	11110011; (1);
9. 10111111;	00110111;	00111011;	10110011; (3);
10. 10101111;	00100111;	00101011;	10100011; (5);
11. 10101011;	00100011;	00101111;	10100111; (7);
12. 10101010;	00100010;	00101110;	10100110; (0);
13. 00101010;	10100010;	10101110;	00100110; (3);
14. 00111010;	10110010;	10111110;	00110110; (5);
15. 00111110;	10110110;	10111010;	00110010; (6);
16. 00111100;	10110100;	10111000;	00110000; (1);
17. 01111100;	11110100;	11111000;	01110000; (3);
18. 01101100;	11100100;	11101000;	01100000; (5);
19. 01101000;	11100000;	11101100;	01100100; (7);
20. 01101001;	11100001;	11101101;	01100101; (0);
21. 11101001;	01100001;	01101101;	11100101; (2);
22. 11001001;	01000001;	01001101;	11000101; (4);
23. 11000001;	01001001;	01000101;	11001101; (6);
24. 11000011;	01001011;	01000111;	11001111; (1);
25. 10000011;	00001011;	00000111;	10001111; (3);
26. 10010011;	00011011;	00010111;	10011111; (5);
27. 10010111;	00011111;	00010011;	10011011; (7);
28. 10010110;	00011110;	00010010;	10011010; (0);
29. 00010110;	10011110;	10010010;	00011010; (3);
30. 00000110;	10001110;	10000010;	00001010; (4);
31. 00001110;	10000110;	10001010;	00000010; (7);
32. 00001111;	10000111;	10001011;	00000011; (1);
33. 01001111;	11000111;	11001011;	01000011; (3);
34. 01011111;	11010111;	11011011;	01010011; (5);
35. 01011011;	11010011;	11011111;	01010111; (7);
36. 01011010;	11010010;	11011110;	01010110; (0);
37. 11011010;	01010010;	01011110;	11010110; (2);
38. 11111010;	01110010;	01111110;	11110110; (4);
39. 11110010;	01111010;	01110110;	11111110; (6);
40. 11110000;	01111000;	01110100;	11111100; (1);
41. 10110000;	00111000;	00110100;	10111100; (3);
42. 10100000;	00101000;	00100100;	10101100; (5);
43. 10100100;	00101100;	00100000;	10101000; (7);
44. 10100101;	00101101;	00100001;	10101001; (0);

45. 00100101;	10101101;	10100001;	00101001; (3);
46. 00110101;	10111101;	10110001;	00111001; (5);
47. 00110001;	10111001;	10110101;	00111101; (6);
48. 00110011;	10111011;	10110111;	00111111; (1);
49. 01110011;	11111011;	11110111;	01111111; (3);
50. 01100011;	11101011;	11100111;	01101111; (5);
51. 01100111;	11101111;	11100011;	01101011; (7);
52. 01100110;	11101110;	11100010;	01101010; (0);
53. 11100110;	01101110;	01100010;	11101010; (2);
54. 11000110;	01001110;	01000010;	11001010; (4);
55. 11001110;	01000110;	01001010;	11000010; (6);
56. 11001100;	01000100;	01001000;	11000000; (1);
57. 10001100;	00000100;	00001000;	10000000; (3);
58. 10011100;	00010100;	00011000;	10010000; (5);
59. 10011000;	00010000;	00011100;	10010100; (7);
60. 10011001;	00010001;	00011101;	10010101; (0);
61. 00011001;	10010001;	10011101;	00010101; (3);
62. 00001001;	10000001;	10001101;	00000101; (4);
63. 00000001;	10001001;	10000101;	00001101; (7).

The list of integers between parentheses in the most right column is the transition sequence of of the four snakes. The translation group

$$\mathcal{G} = <\mathbf{e}_0 + \mathbf{e}_4, \mathbf{e}_0 + \mathbf{e}_5> = \{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_4, \ \mathbf{e}_4 + \mathbf{e}_5, \mathbf{e}_0 + \mathbf{e}_5\}$$

is an invariance group of this cover.

Removing block B_1 from the list provides a basis for a symmetric 4-cover of Q_7 considered as a subgraph of Q_8.

$\mathcal{S}^{(7)}$	$\mathcal{S}^{(7)} + \mathbf{e}_0 + \mathbf{e}_4$	$\mathcal{S}^{(7)} + \mathbf{e}_0 + \mathbf{e}_5$	$\mathcal{S}^{(7)} + \mathbf{e}_4 + \mathbf{e}_5$
0. 00000000;	10001000;	10000100;	00001100; (0);
1. 10000000;	00001000;	00000100;	10001100; (2);
2. 10100000;	00101000;	00100100;	10101100; (4);
3. 10101000;	00100000;	00101100;	10100100; (6);
4. 10101010;	00100010;	00101110;	10100110; (0);
5. 00101010;	10100010;	10101110;	00100110; (3);
6. 00111010;	10110010;	10111110;	00110110; (5);
7. 00111110;	10110110;	10111010;	00110010; (6);
8. 00111100;	10110100;	10111000;	00110000; (0);
9. 10111100;	00110100;	00111000;	10110000; (2);
10. 10011100;	00010100;	00011000;	10010000; (4);
11. 10010100;	00011100;	00010000;	10011000; (6);
12. 10010110;	00011110;	00010010;	10011010; (0);
13. 00010110;	10011110;	10010010;	00011010; (3);
14. 00000110;	10001110;	10000010;	00001010; (4);
15. 00001110;	10000110;	10001010;	00000010; (7);
16. 00001111;	10000111;	10001011;	00000011; (0);
17. 10001111;	00000111;	00001011;	10000011; (2);
18. 10101111;	00100111;	00101011;	10100011; (4);
19. 10100111;	00101111;	00100011;	10101011; (6);
20. 10100101;	00101101;	00100001;	10101001; (0);
21. 00100101;	10101101;	10100001;	00101001; (3);

22. 00110101;	10111101;	10110001;	00111001; (5);
23. 00110001;	10111001;	10110101;	00111101; (6);
24. 00110011;	10111011;	10110111;	00111111; (0);
25. 10110011;	00111011;	00110111;	10111111; (2);
26. 10010011;	00011011;	00010111;	10011111; (4);
27. 10011011;	00010011;	00011111;	10010111; (6);
28. 10011001;	00010001;	00011101;	10010101; (0);
29. 00011001;	10010001;	10011101;	00010101; (3);
30. 00001001;	10000001;	10001101;	00000101; (4);
31. 00000001;	10001001;	10000101;	00001101; (7);

Puncturing with respect to coordinate 1 gives a symmetric 4-cover of Q_7 itself.

We can continue by omitting block B_4 which is the only block containing the integer 7. Now, we obtain a snake $\mathcal{S}^{(6)}$ in Q_6 considered as a subgraph of Q_8 and generated by the block list (B_2 , B_3) with $B_2 = (0, 2, 4, 6)$ and $B_3 = (0, 3, 5, 6)$.

$\mathcal{S}^{(6)}$	$\mathcal{S}^{(6)} + \mathbf{e}_0 + \mathbf{e}_4$	$\mathcal{S}^{(6)} + \mathbf{e}_0 + \mathbf{e}_5$	$\mathcal{S}^{(6)} + \mathbf{e}_4 + \mathbf{e}_5$,
0. 00000000;	10001000;	10000100;	00001100; (0);
1. 10000000;	00001000;	00000100;	10001100; (2);
2. 10100000;	00101000;	00100100;	10101100; (4);
3. 10101000;	00100000;	00101100;	10100100; (6);
4. 10101010;	00100010;	00101110;	10100110; (0);
5. 00101010;	10100010;	10101110;	00100110; (3);
6. 00111010;	10110010;	10111110;	00110110; (5);
7. 00111110;	10110110;	10111010;	00110010; (6);
8. 00111100;	10110100;	10111000;	00110000; (0);
9. 10111100;	00110100;	00111000;	10110000; (2);
10. 10011100;	00010100;	00011000;	10010000; (4);
11. 10010100;	00011100;	00010000;	10011000; (6);
12. 10010110;	00011110;	00010010;	10011010; (0);
13. 00010110;	10011110;	10010010;	00011010; (3);
14. 00000110;	10001110;	10000010;	00001010; (5);
15. 00000010;	10001010;	10000110;	00001110; (6);

The first 15 words of $\mathcal{S}^{(6)}$ and their translations are exactly the same as the corresponding words of $\mathcal{S}^{(7)}$ and their translations. Again, the translation group $\mathbf{G}$ is an invariance group of this cover.

Finally, we can omit one of the two remaining blocks, say B_3, and the remaining block B_2 provides a snake $\mathcal{S}^{(5)}$ with transition sequence

$$0, 2, 4, 6, 0, 2, 4, 6.$$

Actually, this snake is in a subgraph of Q_8 equivalent to Q_4. Translating over the vectors of the group $\mathcal{G} = \{\mathbf{0}, \mathbf{e}_0 + \mathbf{e}_4, \mathbf{e}_4 + \mathbf{e}_5, \mathbf{e}_0 + \mathbf{e}_5\}$ gives the following 4-cover of Q_5.

$\mathcal{S}^{(5)}$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_4$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_5$	$\mathcal{S}^{(5)} + \mathbf{e}_4 + \mathbf{e}_5$
0. 00000000;	10001000;	10000100;	00001100; (0);
1. 10000000;	00001000;	00000100;	10001100; (2);
2. 10100000;	00101000;	00100100;	10101100; (4);
3. 10101000;	00100000;	00101100;	10100100; (6);
4. 10101010;	00100010;	00101110;	10100110; (0);
5. 00101010;	10100010;	10101110;	00100110; (2);
6. 00001010;	10000010;	10001110;	00000110; (4);
7. 00000010;	10001010;	10000110;	00001110; (6).

Puncturing with respect to coordinates 1, 7 and 3 gives a symmetric 4-cover of Q_5 itself. For the sake of completeness we present this 4-cover below.

$\mathcal{S}^{(5)}$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_2$	$\mathcal{S}^{(5)} + \mathbf{e}_0 + \mathbf{e}_3$	$\mathcal{S}^{(5)} + \mathbf{e}_2 + \mathbf{e}_3$
0. 00000;	10100;	10010;	00110;
1. 10000;	00100;	00010;	10110;
2. 11000;	01100;	01010;	11110;
3. 11100;	01000;	01110;	11010;
4. 11101;	01001;	01111;	11011;
5. 01101;	11001;	11111;	01011;
6. 00101;	10001;	10111;	00011;
7. 00001;	10101;	10011;	00111;

Here, we also adjusted the labeling of the coordinate vectors.

Appendix C

A Gray Code of length 8 Related to a 4-Cover of Q_8

If the block list $\mathcal{B} = (B_1, B_3, B_3, B_4)$ of Fig. 7.1 (cf. also Appendix B) with

$$B_1 = (1, 3, 5, 7), \quad B_2 = (0, 2, 4, 6), \quad B_3 = (0, 3, 5, 6), \quad B_4 = (0, 3, 4, 7),$$

is extended by adding two more blocks

$$B_5 = (0, 3, 0, 7), \quad B_6 = (0, 3, 5, 7),$$

the resulting code is a complete Gray code, which is a concatenation of the following four open snakes.

0. 00000000; (1);	64. 10001000; (1);	128. 10000100; (1);	192. 00001100; (1);
1. 01000000; (3);	65. 11001000; (3);	129. 11000100; (3);	193. 01001100; (3);
2. 01010000; (5);	66. 11011000; (5);	130. 11010100; (5);	194. 01011100; (5);
3. 01010100; (7);	67. 11011100; (7);	131. 11010000; (7);	195. 01011000; (7);
4. 01010101; (0);	68. 11011101; (0);	132. 11010001; (0);	196. 01011001; (0);
5. 11010101; (2);	69. 01011101; (2);	133. 01010001; (2);	197. 11011001; (2);
6. 11110101; (4);	70. 01111101; (4);	134. 01110001; (4);	198. 11111001; (4);
7. 11111101; (6);	71. 01110101; (6);	135. 01111001; (6);	199. 11110001; (6);
8. 11111111; (1);	72. 01110111; (1);	136. 01111011; (1);	200. 11110011; (1);
9. 10111111; (3);	73. 00110111; (3);	137. 00111011; (3);	201. 10110011; (3);
10. 10101111; (5);	74. 00100111; (5);	138. 00101011; (5);	202. 10100011; (5);
11. 10101011; (7);	75. 00100011; (7);	139. 00101111; (7);	203. 10100111; (7);
12. 10101010; (0);	76. 00100010; (0);	140. 00101110; (0);	204. 10100110; (0);
13. 00101010; (3);	77. 10100010; (3);	141. 10101110; (3);	205. 00100110; (3);
14. 00111010; (5);	78. 10110010; (5);	142. 10111110; (5);	206. 00110110; (5);
15. 00111110; (6);	79. 10110110; (6);	143. 10111010; (6);	207. 00110010; (6);
16. 00111100; (1);	80. 10110100; (1);	144. 10111000; (1);	208. 00110000; (1);
17. 01111100; (3);	81. 11110100; (3);	145. 11111000; (3);	209. 01110000; (3);
18. 01101100; (5);	82. 11100100; (5);	146. 11101000; (5);	210. 01100000; (5);
19. 01101000; (7);	83. 11100000; (7);	147. 11101100; (7);	211. 01100100; (7);
20. 01101001; (0);	84. 11100001; (0);	148. 11101101; (0);	212. 01100101; (0);
21. 11101001; (2);	85. 01100001; (2);	149. 01101101; (2);	213. 11100101; (2);
22. 11001001; (4);	86. 01000001; (4);	150. 01001101; (4);	214. 11000101; (4);
23. 11000001; (6);	87. 01001001; (6);	151. 01000101; (6);	215. 11001101; (6);
24. 11000011; (1);	88. 01001011; (1);	152. 01000111; (1);	216. 11001111; (1);
25. 10000011; (3);	89. 00001011; (3);	153. 00000111; (3);	217. 10001111; (3);
26. 10010011; (5);	90. 00011011; (5);	154. 00010111; (5);	218. 10011111; (5);
27. 10010111; (7);	91. 00011111; (7);	155. 00010011; (7);	219. 10011011; (7);
28. 10010110; (0);	92. 00011110; (0);	156. 00010010; (0);	220. 10011010; (0);
29. 00010110; (3);	93. 10011110; (3);	157. 10010010; (3);	221. 00011010; (3);
30. 00000110; (4);	94. 10001110; (4);	158. 10000010; (4);	222. 00001010; (4);
31. 00001110; (7);	95. 10000110; (7);	159. 10001010; (7);	223. 00000010; (7);
32. 00001111; (1);	96. 10000111; (1);	160. 10001011; (1);	224. 00000011; (1);
33. 01001111; (3);	97. 11000111; (3);	161. 11001011; (3);	225. 01000011; (3);
34. 01011111; (5);	98. 11010111; (5);	162. 11011011; (5);	226. 01010011; (5);
35. 01011011; (7);	99. 11010011; (7);	163. 11011111; (7);	227. 01010111; (7);
36. 01011010; (0);	100. 11010010; (0);	164. 11011110; (0);	228. 01010110; (0);
37. 11011010; (2);	101. 01010010; (2);	165. 01011110; (2);	229. 11010110; (2);
38. 11111010; (4);	102. 01110010; (4);	166. 01111110; (4);	230. 11110110; (4);
39. 11110010; (6);	103. 01111010; (6);	167. 01110110; (6);	231. 11111110; (6);
40. 11110000; (1);	104. 01111000; (1);	168. 01110100; (1);	232. 11111100; (1);
41. 10110000; (3);	105. 00111000; (3);	169. 00110100; (3);	233. 10111100; (3);
42. 10100000; (5);	106. 00101000; (5);	170. 00100100; (5);	234. 10101100; (5);

43. 10100100; (7);	107. 00101100; (7);	171. 00100000; (7);	235. 10101000; (7);
44. 10100101; (0);	108. 00101101; (0);	172. 00100001; (0);	236. 10101001; (0);
45. 00100101; (3);	109. 10101101; (3);	173. 10100001; (3);	237. 00101001; (3);
46. 00110101; (5);	110. 10111101; (5);	174. 10110001; (5);	238. 00111001; (5);
47. 00110001; (6);	111. 10111001; (6);	175. 10110101; (6);	239. 00111101; (6);
48. 00110011; (1);	112. 10111011; (1);	176. 10110111; (1);	240. 00111111; (1);
49. 01110011; (3);	113. 11111011; (3);	177. 11110111; (3);	241. 01111111; (3);
50. 01100011; (5);	114. 11101011; (5);	178. 11100111; (5);	242. 01101111; (5);
51. 01100111; (7);	115. 11101111; (7);	179. 11100011; (7);	243. 01101011; (7);
52. 01100110; (0);	116. 11101110; (0);	180. 11100010; (0);	244. 01101010; (0);
53. 11100110; (2);	117. 01101110; (2);	181. 01100010; (2);	245. 11101010; (2);
54. 11000110; (4);	118. 01001110; (4);	182. 01000010; (4);	246. 11001010; (4);
55. 11001110; (6);	119. 01000110; (6);	183. 01001010; (6);	247. 11000010; (6);
56. 11001100; (1);	120. 01000100; (1);	184. 01001000; (1);	248. 11000000; (1);
57. 10001100; (3);	121. 00000100; (3);	185. 00001000; (3);	249. 10000000; (3);
58. 10011100; (5);	122. 00010100; (5);	186. 00011000; (5);	250. 10010000; (5);
59. 10011000; (7);	123. 00010000; (7);	187. 00011100; (7);	251. 10010100; (7);
60. 10011001; (0);	124. 00010001; (0);	188. 00011101; (0);	252. 10010101; (0);
61. 00011001; (3);	125. 10010001; (3);	189. 10011101; (3);	253. 00010101; (3);
62. 00001001; (0);	126. 10000001; (5);	190. 10001101; (0);	254. 00000101; (5);
63. 10001001; (7);	127. 10000101; (7);	191. 00001101; (7);	255. 00000001; (7);

By carrying out the cyclic permutation ($\mathbf{w}_{63}$, $\mathbf{w}_{127}$, $\mathbf{w}_{191}$, $\mathbf{w}_{255}$) one obtains the four disjoint snakes which together constitute the symmetric 4-cover of Q_8 presented in the beginning of Appendix B.

Appendix D

A Block List For $R(4, 6)$ code

The following block list $\mathcal{B}$ corresponds to the underlying $R(4, 6)$ code of a snake.

$B_1 = (\underline{15}, 16, 32, 63)$,
$B_2 = (\underline{14}, 16, 32, 62)$,
$B_3 = (\underline{13}, 16, 32, 61)$,
$B_4 = (\underline{12}, 16, 32, 60)$,
$B_5 = (\underline{11}, 16, 32, 59)$,
$B_6 = (\underline{10}, 16, 32, 58)$,
$B_7 = (\ \underline{9}, 16, 32, 57)$,
$B_8 = (\ \underline{8}, 16, 32, 56)$,
$B_9 = (\ \underline{7}, 16, 32, 55)$,
$B_{10} = (\ \underline{6}, 16, 32, 54)$,
$B_{11} = (\ \underline{5}, 16, 32, 53)$,
$B_{12} = (\ \underline{4}, 16, 32, 52)$,
$B_{13} = (\ \underline{3}, 16, 32, 51)$,
$B_{14} = (\ \underline{2}, 16, 32, 50)$,
$B_{15} = (\ \underline{1}, 16, 32, 49)$,
$B_{16} = (\ 0, \underline{16}, 32, 48)$,
$B_{17} = (\ 0, \underline{17}, 23, 49)$.
$B_{18} = (\ 0, \underline{18}, 32, 50)$,
$B_{19} = (\ 0, \underline{19}, 32, 51)$,
$B_{20} = (\ 0, \underline{20}, 32, 52)$,
$B_{21} = (\ 0, \underline{21}, 32, 53)$,
$B_{22} = (\ 0, \underline{22}, 32, 54)$,
$B_{23} = (\ 0, \underline{24}, 32, 56)$.
$B_{24} = (\ 0, \underline{25}, 32, 57)$,
$B_{25} = (\ 0, \underline{26}, 32, 58)$,
$B_{26} = (\ 0, \underline{28}, 32, 60)$,
$B_{27} = (\ 0, 31, 32, \underline{63})$,
$B_{28} = (\ 0, 30, \underline{32}, 62)$.
$B_{29} = (\ 0, 31, 34, \underline{61})$,
$B_{30} = (\ 0, 30, \underline{34}, 60)$,
$B_{31} = (\ 0, 31, 36, \underline{59})$,
$B_{32} = (\ 0, 30, \underline{36}, 58)$,
$B_{33} = (\ 0, 31, 38, \underline{57})$,
$B_{34} = (\ 0, 30, \underline{38}, 56)$.
$B_{35} = (\ 0, 31, 40, \underline{55})$,
$B_{36} = (\ 0, 30, \underline{40}, 54)$,
$B_{37} = (\ 0, 31, 42, \underline{53})$,
$B_{38} = (\ 0, 30, \underline{42}, 52)$,
$B_{39} = (\ 0, 31, 44, \underline{51})$,
$B_{40} = (\ 0, 30, \underline{44}, 50)$,
$B_{41} = (\ 0, 31, 46, \underline{49})$,
$B_{42} = (\ 0, \underline{30}, 46, 48)$,

$B_{43} = (\ 0, 31, 33, \underline{62})$,
$B_{44} = (\ 0, 29, \underline{33}, 60)$.
$B_{45} = (\ 0, 31, 37, \underline{58})$,
$B_{46} = (\ 0, 29, \underline{37}, 56)$,
$B_{47} = (\ 0, 31, 41, \underline{54})$,
$B_{48} = (\ 0, 29, \underline{41}, 52)$,
$B_{49} = (\ 0, 31, 45, \underline{50})$,
$B_{50} = (\ 0, \underline{29}, 45, 48)$.
$B_{51} = (\ 0, 31, 35, \underline{60})$,
$B_{52} = (\ 0, 27, \underline{35}, 56)$,
$B_{53} = (\ 0, 31, 43, \underline{52})$,
$B_{54} = (\ 0, \underline{27}, 43, 48)$,
$B_{55} = (\ 0, 31, 39, \underline{56})$,
$B_{56} = (\ 0, \underline{23}, 39, 48)$,
$B_{57} = (\ 0, 31, \underline{47}, 48)$.

The additional blocks, to extend $\mathcal{B}$ to $\mathcal{B}^{ext}$, consist of the block

$$B_{58} = (0, 31, 0, 48)$$

together with the following four blocks

$B_{59} = (0, 31, 46, 48)$, $B_{60} = (0, 31, 45, 48)$, $B_{61} = (0, 31, 43, 48)$, $B_{62} = (0, 31, 39, 48)$.

List of References

[1]. Abbott, H.L., Katchalski, M, On the Construction of snake-in-the-box codes, *Utilitas Math*. **40** (1991), 97 - 116.

[2]. Alsardary, S.Y., Further Results on Vertex Covering of Powers of Complete Graphs, *Acta. Math. Univ. Comenianae,* **LXVI**, 2 (1997), 261 - 283.

[3]. Biggs, N.L., White, A.T., Permutation Groups and Combinatorial Structures, Cambridge University Press (1979).

[4]. Casella, D.A., New Lower Bounds for the Snake-in-the-Box and the Coil-in-the-Box Problems: Using Evolutionary Techniques to Hunt for Snakes and Coils, MSAI Thesis, (2005).

[5]. Casella, D.A. and Potter, W.D., New Lower Bounds for the Snake-In-The-Box Problem: Using Evolutionary Techniques to Hunt for Snakes, *Proc. of the 18th Int. Florida Artificial Intelligence Research Conference*, (2005), 264 - 269

[6]. Diaz-Gomez, P.A., Hougen, D.F., The snake in the box problem: mathematical conjecture and a genetic algorithm approach, *Proc.of the 8th Annual Conference on Genetic and Evolutionary Computation, Seattle*, Washington, July 08 - 12, 2006.

[7]. Emelyanov, P. G., A. An Upper-Bound for the Length of a Snake in the *n*-Dimensional Unit Cube, in Discrete Analysis and Operations Researchs (ed. A.D. Kurshunov), Kluwer Academic Publisher, Dordecht/Boston/London, 1996.

[8]. Emelyanov, P. G., Lukito, A., On the maximal length of a snake in a hypercube of small dimension, *Discrete Math*. **218** (2000), 51-59.

[9]. Evdokimov, A.A, Circuit codes with arbitrary distance, *Soviet Math. Dokl.,* **17** (1976), 900 - 904.

[10]. Evdokimov, A.A, Numeration of subsets of a finite set. (Russian), *Metody Diskret. Analiz*. **34** (1980), 8-26.

[11]. Haryanto, L., Zanten, A.J.van, Snakes-in-the-box Codes and Euclidean Geometry, *Proc. of the* 9^{th} *Int. Workshop on Algebraic and Combinatorial Coding Theory*, June 19-25, 2004, Kranevo, Bulgaria.

[12]. Hiltgen, A.P., Paterson, K.G., Brandestini, M., (1996). Single-Track Gray Codes. *IEEE Trans. Inform. Theory*, vol. 42, No.5, 779-789.

[13]. Hiltgen, A.P., Paterson, K.G., (2001). Single-Track Circuit Codes. *IEEE Trans. Inform. Theory,* vol. 47, No.6, 2587-2595.

[14]. Huffman, W.C., Pless, V., Fundamentals of Error-Correcting Codes, Cambridge University Press (2003).

[15]. Kautz, W.H., Unit Distance Error Checking Codes, *IEEE Trans. Electron. Compute.*, **7** (1958), 179-180.

[16]. Klee, V., A Method for Constructing Circuit Codes, *J. Assoc. Comput. Mach*. **14** (1967), 520-528.

[17]. Kochut, K.J., Snake-in-the-box Codes for Dimension 7, *J. Combin.. Math*. *Combin. Comput.* **20** (1996), 175 - 185.

[18]. Knuth, D.E., The Art of Computer Programming, Volume 4 Fascicle 2, Generating all tuples and permutations, Addison Wesley (2005).

[19]. Lukito, A., Bounds for the Length of Certain Types of Distance Preserving Codes, Ph.D. Thesis, Delft University of Technology, 2000.

[20]. Lukito, A., van Zanten, A.J., A new non-asymptotic upper bound for snake-in-the-box codes, *J. Combin. Math. Combin. Comput*., **39** (2001), 147 – 156..

[21]. Lukito, A., van Zanten, A.J., A new non-asymptotic upper bound for snake-in-the-box codes, *Proc. of the 7th Int. Workshop on Algebraic and Combinatorial Coding Theory*, Bansko, Bulgaria (2000).

[22]. Logacev, V.N., An Improvement of the Griesmer Bound in the Case of Small Code Distances, Optimization Methods and their Applications (All-Union Summer Sem., Khakusy, Lake Baikal (Russian)), **182** (1972), 107-111.

[23]. MacWilliams, F.J., Sloane, N.J.A., The Theory of Error Correcting Codes. North Holland Mathematical Library, 1977.

[24]. Paterson, K.G., Tuliani, G., Some New Cicuit Codes, *IEEE Trans. Inf. Theory*, **44** (1998), 1305-1309

[25]. Preparata, F., and Nievergelt, J. Difference-preserving codes, *IEEE Trans. Inform. Theory*, **20** (1974), 643-649.

[26]. Potter, W.D, et al, Using the Genetic Algorithm to Find Snake-in-the-bpx Codes, *Proc. of the 7th Int. Conf. on Industrial and Engin. Applic. of artificial Intell. and Expert Systems*, Austin, Texas, United States (1994), 421 - 426.

[27]. Reingold, E.M., Nievergelt, J., N.Deo, *Combinatorial Algorithms: Theory and Practice*, Prentice Hall (1977)

[28]. Schwartz, M., Etzion, T., The Structure of Single-Track Gray Codes. *IEEE Trans. Inform. Theory*, **45** (1999), 2383-2396.

[29]. Simonis, J., On Generator Matrices of Codes, *IEEE Trans. Inf. Theory*, **38** (1992), 516.

[30]. Snevily, H.S., The Snake-in-the-box problem, A new upper bound, *Discrete Math.* **133** (1994), 307 - 304.

[31]. Solov'jeva, F.I., An upper bound for the length of a cycle in an *n*-dimensional unit cube, *Diskret. Analiz.* **45** (1987), p. 71-76.

[32]. Wojciechowski, J., Covering the Hypercube with a Bounded Number of Disjoint Snakes, *Combinatorica*, **14** (1994), 491-496.

[33]. Zanten, A.J.van, Haryanto, L., Snakes and Necklaces, Report CS 02-08, Department of Computer Science, Universiteit Maastricht, 2002.

[34]. Zanten, A.J.van, Haryanto, L., On The Transition Sequences of Symmetric Snakes, Report CS 03-04, Department of Computer Science, Universiteit Maastricht, 2003.

[35]. Zanten, A.J.van, Haryanto, L., Covers of Hypercubes by Snakes and Euclidean Geometries, Report CS 04-03, Department of Computer Science, Universiteit Maastricht, 2004.

[36]. Zanten, A.J.van, Haryanto, L., Covers and Near-Covers of the Hypercube Q_{16} by Symmetric Snakes, Report CS 06-01, Department of Computer Science, Universiteit Maastricht, 2006.

[37]. Zanten, A.J.van, Haryanto, L., A 2^{m-1}-Cover of Snakes for Q_n, $n = 2^m$, Report MICC 06 - 03, Department of Computer Science, Universiteit Maastricht, 2006.

[38]. Zanten, A.J.van, Haryanto, L., Sets of Disjoint Snakes Based on a Reed-Muller Code and Covering the Hypercube, submitted to *Designs, Codes and Cryptography*.

[39]. Zanten, A.J. van, Lukito, A., Construction of Certain Cyclic Distance-Preserving Codes Having Linear-Algebraic Characteristic, *Designs, Codes and Cryptography*, **16** (1999), 185-199.

[40]. Zanten, A.J. van, Lukito, A., Covers of Hypercubes with Symmetric Snakes, Report CS00-04, Department of Computer Science, Universiteit Maastricht, 2000.

[41]. Zanten, A.J. van, Lukito, A., Vertex Partitions of Hypercubes into Symmetric Snakes, *Electronic Notes in Discrete Mathematics*, **11** (2002).

[42]. Zémor, G., An Upper Bound of the Size of Snakes, *Combinatorica*, **17** (1997), 287-298.

Summary

A snake-in-the-box code (or snake) is a list of binary words of length n such that each word differs from its successor in the list in precisely one bit position. Moreover, any two words in the list differ in at least two positions, unless they are neighbours in the list. The list is considered to be a cyclic list which implies that the 'first word' is the successor of the 'last word'. So, actually there is no first or last word. The two defining rules mentioned above have to be interpreted in cylic sense.

There are some methods known to construct such codes. These are all of a recursive nature, i.e. snakes of a certain length n are constructed from snakes of length $m < n$. These snakes of length m are already known in some way, either by similar recursive methods, or by computer search, or just by accident.

In Chapter 3 of this thesis such a known recursive technique is discussed and slightly generalized. In the remaining part of the thesis a new method is developed to construct snakes in a straightforward, i.e. non-recursive, way. Since this method uses a linear algebraic code, the constructed snakes have an additional kind of structure apart from their 'snake structure', More precisely, every fourth word of the snake is a word of the applied linear code, and hence, part of the snake (its 'backbone') has the structure of a linear space. Due to this linearity, we are able to give a partial answer to an old problem posed by Erdös: how many disjoint snakes of the same length are sufficient to contain all binary words of length n?

For $n = 2$, this number is equal to 1, and for $2 < n \leq 4$, it is equal to 2. We found that 4 snakes suffice for $4 < n \leq 8$, and 8 snakes for $8 < n \leq 16$, which improves the results known thus far. For $16 < n \leq 32$ we find a cover of 16 snakes. This number 16 is equal to the most recent result in the literature. However, since the snakes themselves have a nice symmetric property, and are constructed in a straightforward way, we claim that this result is also better than the results known in the literature. For $n > 32$, the number of snakes in a cover constructed with our method grows far beyond the upper bound of 16 proven by Wojciechowski. However, contrary to the snakes and covers in his proof which are only proven to exist, ours are concrete and possess the additional structure and symmetry already mentioned.

Samenvatting

Een slangcode (ofwel slang) is een lijst van binaire woorden van lengte n, zodanig dat elk woord in precies één bitpositie verschilt van zijn opvolger in de lijst. Bovendien verschillen elke twee woorden in de lijst in tenminste twee bitposities, tenzij ze buren van elkaar zijn in de lijst. De lijst wordt beschouwd als een cyclische lijst, hetgeen impliceert dat het 'eerste woord' de opvolger is van het 'laatste woord'. Eigenlijk is er dus geen eerste of laatste woord. De twee hierboven genoemde definitieregels moeten in cyclische zin geïnterpreteerd worden. Er is een aantal methoden bekend om zulke codes te construeren. Deze zijn alle van recursieve aard, dat wil zeggen dat slangen van een zekere woordlengte n geconstrueerd worden uit slangen met woordlengte $m < n$. De slangen van woordlengte m, op hun beurt, zijn dan op een of andere manier al bekend, ofwel door middel van een soortgelijke recursieve techniek, of via een zoekprograma of gewoon bij toeval.

Na de inleidende hoofdstukken 1 en 2, wordt in hoofdstuk 3 van dit proefschrift zo'n recursieve methode besproken en enigszins gegeneraliseerd. In het resterende deel van het proefschrift wordt een nieuw methode ontwikkeld om slangen te construeren op een rechtstreekse, dat wil zeggen niet-recursieve, manier. Omdat in onze methode een lineaire algebraische code wordt gebruikt, hebben de geconstrueerde slangen een extra soort struktuur bovenop hun eigenlijke 'slangstruktuur'. Iets preciezer gezegd, elk vierde woord van de slang is een woord van de gebruikte lineaire code en dus heeft een gedeelte van de slang (de 'rugge-graat') de struktuur van een lineaire ruimte. Dank zij deze lineariteit zijn we in staat om een gedeeltelijk antwoord te geven op een oud probleem van Erdös: Hoeveel disjuncte slangen van dezelfde (lijst-)lengte (overdekking) zijn voldoende om alle binaire woorden van lengte n te bevatten? Voor $n = 2$ is dit aantal gelijk aan 1 en voor $2 < n \leq 4$ is het gelijk aan 2. Wij vonden dat 4 slangen voldoende zijn voor $4 < n \leq 8$, en 8 slangen voor $8 \leq n \leq 16$. Deze aantallen zijn beter dan de resultaten die tot dusver bekend waren. Voor $16 < n \leq 32$, vonden we een overdekking bestaande uit 16 slangen. Dit getal van 16 is gelijk aan het meest recente resultaat in de literatuur. Omdat de slangen zelf echter op een rechtstreekse en concrete manier zijn geconstrueerd en ook een zekere symmetrie bezitten, claimen we dat ook dit resultaat een verbetering is. Voor $n > 32$ wordt het aantal slangen in de door ons geconstrueerde overdekkingen veel groter dan de door Wojciechowski bewezen bovengrens van 16. In tegenstelling echter tot de slangen en overdekkingen in zijn bewijs waarvan slechts het bestaan wordt aangetoond, zijn de door ons geconstrueerde concreet en bezitten tevens de reeds genoemde extra struktuur en symmetrie.

Curriculum Vitae

Loeky Haryanto was born in Solo, Central Java, Indonesia, on September 15, 1955. The author obtained his bachelor degree in 1982 at the Mathematics Department, Gadjah Mada University, Yogyakarta, Indonesia, after writing a final paper titled *Measure Theory*. Since 1983, the author has taught introductory mathematics for undergraduate students at the Mathematics Department of Hasanuddin University, Makassar, in Indonesia, like Abstract Algebra, Linear Algebra, Real Analysis, Logic, Probability and Statistics. In 1988, he obtained his first master degree at the Bandung Institute of Technology, Indonesia, on a paper titled *Martingale(s)*.

In 1989, the author was selected to join an extensive training for an overseas fellowship program conducted by the Indonesian government and financed by the World Bank. As a result, the author was admitted (1990) at the Mathematics Department of the University of Florida (UF), the United States. After following courses at UF from Spring 1991 until Fall 1993, he was awarded with a Master of Science and a Master of Arts in Teaching, both with a major in mathematics.

In 2001, he was selected as the best participant of the Workshop on Algebraic Coding Theory and Cryptography at Bandung Institute of Technology. As a follow up, he was given the opportunity to do research at Technische Universiteit Delft (TUDelft) for a Ph.D degree, under the supervision of Prof. dr.A.J. van Zanten and with financial support of the Koninklijke Nederlandse Academie van Wetenschappen (KNAW).

Prior to his professional career as a lecturer at the Hasanuddin University, Makassar, on the island of Sulawesi (Celebes), and apart from his higher education on the island of Java and abroad, his early education was mostly accomplished in Banjarmasin, Kalimantan (Borneo), where he lived for twelve years. When he has completed his Ph.D study in the Netherlands, he will resume teaching students and take up his life again with his wife and two children in Makassar, Sulawesi, Indonesia.

List of Publications

This thesis is based on the following publications which were written during the period July 2002 – December 2006.

1. Haryanto, L., Zanten, A.J.van, Snakes-in-the-box Codes and Euclidean Geometry, *Proc. of the* 9^{th} *Int. Workshop on Algebraic and Combinatorial Coding Theory*, June 19-25, 2004, Kranevo, Bulgaria.
2. Zanten, A.J.van, Haryanto, L., Snakes and Necklaces, Report CS 02-08, Department of Computer Science, Universiteit Maastricht, 2002.
3. Zanten, A.J.van, Haryanto, L., On The Transition Sequences of Symmetric Snakes, Report CS 03-04, Department of Computer Science, Universiteit Maastricht, 2003.
4. Zanten, A.J.van, Haryanto, L., Covers of Hypercubes by Snakes and Euclidean Geometries, Report CS 04-03, Department of Computer Science, Universiteit Maastricht, 2004.
5. Zanten, A.J.van, Haryanto, L., Covers and Near-Covers of the Hypercube Q_{16} by Symmetric Snakes, Report CS 06-01, Department of Computer Science, Universiteit Maastricht, 2006.
6. Zanten, A.J.van, Haryanto, L., A 2^{m-1}-Cover of Snakes for Q_n, $n = 2^m$, Report MICC 06 - 03, Department of Computer Science, Universiteit Maastricht, 2006.
7. Zanten, A.J.van, Haryanto, L., Sets of Disjoint Snakes Based on a Reed-Muller Code and Covering the Hypercube, submitted to *Designs, Codes and Cryptography*.

Index