Structural Design and Analysis of Low-Density Parity-Check Codes and Systematic Repeat-Accumulate Codes

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

Alexander Gruner

aus Reutlingen

Tübingen, 2015

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	12.06.2015
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Prof. Dr. Michael Huber
2. Berichterstatter:	Prof. Dr. Peter Hauck

Inhaltsverzeichnis

Abstract

The discovery of two fundamental error-correcting code families, known as turbo codes and low-density parity-check (LDPC) codes, has led to a revolution in coding theory and to a paradigm shift from traditional algebraic codes towards modern graph-based codes that can be decoded by iterative message passing algorithms. From then on, it has become a focal point of research to develop powerful LDPC and turbo-like codes. Besides the classical domain of randomly constructed codes, an alternative and competitive line of research is concerned with highly structured LDPC and turbo-like codes based on combinatorial designs. Such codes are typically characterized by high code rates already at small to moderate code lengths and good code properties such as the avoidance of harmful 4-cycles in the code's factor graph. Furthermore, their structure can usually be exploited for an efficient implementation, in particular, they can be encoded with low complexity as opposed to random-like codes. Hence, these codes are suitable for high-speed applications such as magnetic recording or optical communication.

This thesis greatly contributes to the field of structured LDPC codes and systematic repeat-accumulate (sRA) codes as a subclass of turbo-like codes by presenting new combinatorial construction techniques and algebraic methods for an improved code design. More specifically, novel and infinite families of high-rate structured LDPC codes and sRA codes are presented based on balanced incomplete block designs (BIBDs), which form a subclass of combinatorial designs. Besides of showing excellent error-correcting capabilites under iterative decoding, these codes can be implemented efficiently, since their inner structure enables low-complexity encoding and accelerated decoding algorithms.

A further infinite series of structured LDPC codes is presented based on the notion of transversal designs, which form another subclass of combinatorial designs. By a proper configuration of these codes, they reveal an excellent decoding performance under iterative decoding, in particular, with very low error-floors. The approach for lowering these error-floors is threefold. First, a thorough analysis of the decoding failures is carried out, resulting in an extensive classification of so-called stopping sets and absorbing sets. These combinatorial entities are known to be the main cause of decoding failures in the error-floor region over the binary erasure channel (BEC) and additive white Gaussian noise (AWGN) channel, respectively. Second, the specific code structures are exploited in order to calculate conditions for the avoidance of the most harmful stopping and absorbing sets. Third, powerful design strategies are derived for the identification of those code instances with the best error-floor performances. The resulting codes can additionally be encoded with low complexity and thus are ideally suited for practical high-speed applications.

Further investigations are carried out on the infinite family of structured LDPC codes based on finite geometries. It is known that these codes perform very well under iterative decoding and that their encoding can be achieved with low complexity. By combining the latest findings in the fields of finite geometries and combinatorial designs, we generate new theoretical insights about the decoding failures of such codes under iterative decoding. These examinations finally help to identify the geometric codes with the most beneficial error-correcting capabilities over the BEC.

1

Introduction

Many forms of digital communication are visible in daily life such as telephone communication, television, computer networks and radio broadcasting. Besides of that, there are more inconspicuous types of digital communication within the fields of electrical engineering, computer science, aerospace engineering and many more. All these forms of digital communication have in common, that the transmission of information is disturbed by physical interferences such as atmospheric disturbances or hardware failures, leading to errors in the received data. Hence, since the advent of the first data transmission applications, there is a great interest in the detection and correction of errors that occur during the transmission or storage of digital data. Today, we are still facing big challenges in order to keep pace with technological advancements, novel application scenarios as well as new trends in the field of coding theory such as quantum communication. Therefore, new and powerful error-correction techniques have to be invented, explored and developed in the future.

Looking back to the year 1948, Claude E. Shannon has published a groundbreaking paper [1] that gave birth to the fields of information theory and coding theory. In this paper, Shannon formalized the concept of information and established fundamental limits for the maximum amount of information that can potentially be transmitted over an unreliable communication channel [2]. Shannon's efforts culminated in his famous channel coding theorem, which states that for any given channel, there exists an upper

limit, called the channel's capacity, for which reliable transmission is possible for rates smaller than the capacity and reliable transmission is not possible for rates above the capacity. More precisely, there always exists a block code with rate smaller than the channel capacity for which the decoding error probability is arbitrarily small. Unfortunately, the proof of this theorem is non-constructive such that it does not deliver any concrete codes with capacity-achieving decoding performances. As a consequence, it has become a major challenge to construct practical codes capable of achieving the limit of Shannon's coding theorem along with manageable encoding and decoding complexity.

A popular class of capacity-achieving codes are *low-density parity-check* (LDPC) codes that were originally introduced by Gallager in 1962 [3] and further examined in his remarkable doctoral thesis [4]. In this thesis, Gallager also conceived the idea of iterative decoding under which LDPC codes are potentially capable of achieving the limit of Shannon's coding theorem. However, at this time, the implementation of an efficient iterative decoder for LDPC codes was infeasible due to the limited computational power such that the powerfulness of iterative decoding could not be fully demonstrated [5]. As a consequence, the class of LDPC codes was largely ignored for over three decades. Moreover, the field of error-correcting codes was dominated by highly-structured algebraic block codes and convolutional codes such that the enormous potential of LDPC codes has not been recognized since they pursued a completely different coding paradigma.

The discovery of turbo codes in 1993 by Berrou et al. [6, 7] revolutionized the field of coding theory and led to a fundamental paradigm shift from algebraic-driven approaches to modern codes with very long code lengths that can be decoded by iterative message-passing algorithms. By using a parallel concatenation of two simple constituent codes and a pseudo-random block interleaver, Berrou et al. surprisingly discovered codes achieving Shannon's theoretical limits, having enough structure to enable low encoding and decoding algorithms such that these codes are suitable for practical applications. After the rediscovery of LDPC codes by Mackay and Neal in 1995 [8] there has been renewed interest in LDPC codes. Decoded with iterative message-passing algorithms, these co-

des have shown to perform remarkably close to the Shannon limit. Subsequently, LDPC codes have become a focal point of research.

Inspired by the turbo revolution, other families of turbo-like codes has been developed as competitive alternatives to turbo codes and LDPC codes. One example are so-called *repeat-accumulate* (RA) codes introduced in [9]. This code family was primarily developed for theoretical purposes and have an extremly simple encoding scheme that basically consists of a serial concatenation of two simple constituent codes, namely a repetition code and an accumulator. Despite of their simplicity, it has soon been realized that properly designed RA codes exhibit an excellent decoding performance. Also, it has been recognized that the systematic version of RA codes[1], called *systematic RA* (sRA) codes can be interpreted as LDPC codes (see for example, [10], and the references therein). Hence, sRA codes can be simultaneously seen as a class of simple turbo-like codes and a class of LDPC codes [11]. This dual representation can be exploited by using the turbo-like code representation for low-complexity encoding and the LDPC code representation for graph-based iterative decoding, gaining the benefits of both representations.

The codes considered in the current thesis are highly relevant for pratical applications in actual and future generations of communication system standards. More precisely, LDPC codes are adopted in several recent wireless networking standards such as IEEE802.11n (WiFi), IEEE802.16d (WiMAX), IEEE 802.16e (WiMAX) and in 10 Gb/s ethernet IEEE 803.3an (cf. [12, 13] and the references therein), as well as in digital video broadcasting (DVB-2) [14] and many others. Furthermore, they are considered for a wide range of applications with low BER requirements such as optical channels and magnetic disk drives [15, 16]. Also, repeat-accumulate codes are used in today's and future standards such as in DVB-S2 and IEEE802.16 (WirelessMAN).

[1] The systematic version of an RA code means that the orginal message bits are concatenated with the output of the accumulator such that the message is directly embedded in the encoded output.

1.1. Thesis overview

The present thesis is structured as follows. Chapter 2 gives a short overview of the basic concepts of coding theory in Section 2.1 and introduces the family of low-density parity-check (LDPC) codes in Section 2.2 as well as the family of systematic repeat-accumulate (sRA) codes in Section 2.3. For both code families, Section 2.4 gives a short survey of the basic principles of good code design.

Chapter 3 presents new families of structured LDPC codes based on combinatorial construction techniques. Firstly, Section 3.1 gives a short introduction into the concept of balanced incomplete block designs (BIBDs) as a basic notion of the field of combinatorial design theory, and describes their connection to the design of LDPC codes. Secondly, Section 3.2 and 3.3 presents novel classes of high-rate LDPC codes and sRA codes, respectively, based on certain classes of BIBDs. Finally, the error-correcting capabilities of the novel codes are demonstrated by extensive Monte-Carlo simulations in Section 3.4.

Chapter 4 presents new families of LDPC codes based on transversal designs with optimized performances over the BEC and AWGN channel, in particular, with very low error-floors. After shortly introducing the basic concepts of set systems, Latin squares and transversal designs in Section 4.1, it is shown in Section 4.2, how LDPC codes can be constructed from transversal designs with cyclic structure. Based on this specific family of LDPC codes, an extensive stopping set analysis is carried out in Section 4.3, which is then utilized for the design of LDPC codes with highly beneficial stopping set distributions in Section 4.4, leading to codes with very low error-floors over the BEC. Similarly, after conducting an absorbing set analysis in Section 4.5, powerful design strategies are developed in Section 4.6 for the construction of LDPC codes with benefical absorbing set distributions, leading to excellent codes over the AWGN channel. Subsequently, the decoding performances of the arising LDPC codes are demonstrated in Section 4.7 by comprehensive computer simulations over the BEC and AWGN channel. The chapter is closed by an extensive discussion which compares the novel TD LDPC codes to the

BIBD LDPC codes presented in the previous chapter.

Chapter 5 gives some new theoretical insights into the family of LDPC codes based on finite geometries, more specifically, based on projective and affine geometries which are introduced in Section 5.1. Section 5.2 gives a short overview of the different types of LDPC codes that can be constructed based on finite geometries. In Section 5.3, a thorough stopping set analysis is carried out which results in fundamental bounds for the stopping distances of LDPC codes based on finite geometries. Finally, Section 5.4 summarizes and discusses the results of the chapter.

Chapter 6 concludes the thesis by discussing the major results, while Section 6.1 gives an outlook on upcoming tasks that can be followed up.

1.2. Publications and contribution

The present thesis is based on the results of the author's publications [P1], [P2] and the submitted manuscript [P3], but it extends their outcome substantially. Although structured and combinatorial code design is an integral part of the last manuscript [P4], its quantum theoretic coding approach separates it thematically from the other papers such that it is not treated in the present thesis.

Section 1.2.1 gives references to all publications and manuscripts in which the doctoral candidate and author of the present thesis has been involved. Subsequently, Section 1.2.2 describes the candidate's contribution to the listed manuscripts.

1.2.1. Publications and manuscripts

[P1] A. Gruner and M. Huber, "New Combinatorial Construction Techniques for Low-Density Parity-Check Codes and Systematic Repeat-Accumulate Codes," *IEEE Trans. on Communications*, vol. 60, no. 9, pp. 2387–2395, 2012.

[P2] A. Gruner and M. Huber, "Low-Density Parity-Check Codes from Transversal Designs with Improved Stopping Set Distributions," *IEEE Trans. on Communications*, vol. 61, no. 6, pp. 2190–2200, 2013.

[P3] A. Gruner and M. Huber, "Absorbing Set Analysis and Design of LDPC Codes from Transversal Designs over the AWGN Channel," *submitted to IEEE Trans. on Information Theory*, 2014.

[P4] Y. Fujiwara, A. Gruner and P. Vandendriessche, "High-rate quantum low-density parity-check codes assisted by reliable qubits," *accepted for publication in IEEE Trans. on Information Theory*, 2014.

1.2.2. Contribution of the candidate

Regarding the first publication [P1], the doctoral candidate is the lead author and has been responsible for the Sections I, IV, and V, whereas the co-author has been in charge of Sections II and III. The scientific ideas, as well as the analysis and interpretation of the results have been worked out under the collaboration of both authors in equal parts. All simulations have been performed, visualized and interpreted by the candidate based on a self-developed MATLAB software.

For the manuscripts [P2] and [P3], the candidate is the lead author as well as the originator of the scientific ideas of both papers. Furthermore, the data generation and paper writing has been completely carried out by the doctoral candidate. The results have been analyzed and interpreted mainly by the lead author in collaboration with the co-author. Finally, all simulations have been performed, presented an evaluated by the candidate.

For the last manuscript [P4], the candidate is a co-author and has been responsible for the implementation and presentation of all experimental results. Furthermore, he has contributed to the analysis and interpretation of the results.

2

Basics

2.1. Coding theory

The field of coding theory is concerned with the protection of data against physical errors such as atmospheric disturbances and hardware failures when transferring the data over a noisy communication channel or when storing the data in an error-prone storage system. By contrast, the related area of cryptology is concerned with the protection of data against unauthorized wiretapping and data manipulation by third parties (called adversaries). The coding theory is also closely related to the field of information theory and interacts with several scientific disciplines such as discrete and applied mathematics, computer science and electrical engineering.

2.1.1. Error-correcting codes

The basic notion in coding theory is the so-called *error-correcting code* which is used to detect and correct errors. The essential idea of an error-correcting code is to add redundant data systematically to a message so that the receiver can recover the potentially corrupted message by exploiting the code's structure. Fig. 2.1 shows the components of a channel coding system. A sender delivers messages that shall be transmitted over

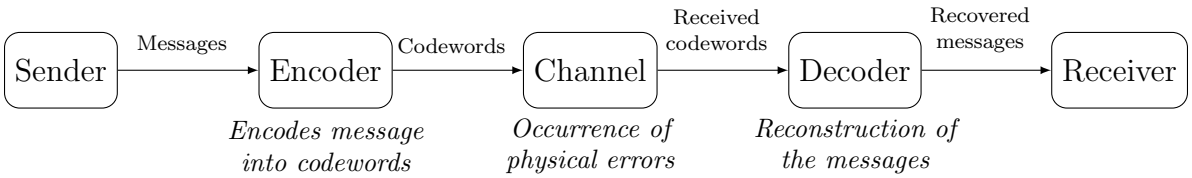


Abbildung 2.1. – Channel coding system

a noisy channel and the encoder translates these messages into codewords containing redundant data. The arising codewords are then transmitted over a channel which may introduce errors due to physical disturbances. The decoder finally attempts to recover the original message by using the redundancy and structure of the received codewords.

2.1.2. Linear block codes

The most important class of error-correcting codes are so-called *block codes* which encode messages of K symbols into codewords (or blocks) of N symbols where $N > K$. A *linear block code*, denoted by $\mathcal{C}$, with *code length* N and *dimension* K is a linear K-dimensional subspace of the vector space $\mathbb{F}_q^N$, where $\mathbb{F}_q$ is the finite field of prime power order q. The vectors of (the subspace) $\mathcal{C}$ are called *codewords*. In this thesis, we only consider binary codes with $q = 2$ such that each codeword is a sequence of N bits. Since linear subspaces can be spanned by a minimal set of basis vectors, every block code $\mathcal{C}$ can be represented by a minimal set of K codewords. By writing these codewords as the columns of an $N \times K$ matrix, we obtain the generator matrix G of the code. The generator matrix provides a way to encode a message $x \in \mathbb{F}_q^K$ (thought as a column vector) into the codeword $Gx \in \mathcal{C}$, such that

$$\mathcal{C} = \{Gx : x \in \mathbb{F}_q^K\}.$$

Furthermore, if for any $M \times N$ matrix H with $M \geq N - K$ holds that

$$\mathcal{C} = \{c \in \mathbb{F}_q^N : Hc = 0\},$$

then H is called a parity-check matrix of the code, i.e., $\mathcal{C}$ is implicitly given by the null

space of H. Since the columns of G are codewords of $\mathcal{C}$, it clearly holds that $HG = 0$. Also note that the parity-check matrix H of a code is not unique, i.e., there are many different representations for a parity-check matrix leading to the same code. It is very important to design the parity-check matrix properly since it influences the code's performance under iterative decoding and furthermore, since specific structures can be exploited for the practical implementation of the code.

The *hamming distance* between two codewords is equal to the number of symbols in which they differ and the *weight* of a codeword is its hamming distance to the all-zero codeword. In the classical domain of algebraic coding theory, an important property of a linear code, $\mathcal{C}$, is its *minimal distance*, denoted by $d_{min}(\mathcal{C})$, which is defined as the smallest hamming distance between any two codewords of $\mathcal{C}$ or, equivalently, as the smallest weight of any codeword of $\mathcal{C}$. The minimal distance of a code plays an essential role for the classical *maximum-likelihood* (ML) decoder, since the ML decoding strategy is to find the codeword with the smallest hamming distance to the received codeword. Obviously, it is guaranteed that a number of $\lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$ or less errors can be corrected.

2.2. Low-density parity-check (LDPC) codes

Whereas in the classical domain of algebraic codes (e.g. [17, 18]), the vector space of their codewords is at the forefront in order to maximize the code's minimum distance under ML decoding, the core part of an LDPC code is its parity-check matrix H. The central idea of an LDPC code [3] is that H is sparse, meaning that the number of ones in H is small compared to the number of zeros. The advantage of a sparse parity-check matrix is that an *iterative decoding algorithm* can efficiently operate on the code's *factor (or Tanner) graph* [19] which is an equivalent representation of the parity-check matrix. More precisely, let $G_H = (V, F, E)$ denote a bipartite graph, where the nodes V are associated with the rows of H, the nodes F are associated with the columns of H and E

is a set of undirected edges such that the i-th node of V is connected to the j-th node of F if and only if $H_{ij} = 1$. The elements of V are called *bit nodes*, since they correspond to the code bits, and the elements of F are called *check nodes*, since they correspond to the parity-check equations of the code.

Classically important parameters of an LDPC code are the code length N, which is equal to the number of columns of H, the dimension $K = N - \text{rank}_2(H)$, the rate $R = 1 - \frac{\text{rank}_2(H)}{N}$ and the design rate $R_d = 1 - \frac{M}{N}$ where M is the number of rows of H. Note that the design rate is equal to the code rate if and only if the 2-rank of H is full, else $R > R_d$. If the parity-check matrix of an LDPC code has constant column weights k and constant row weights r, then the code is called (k, r)-*regular*.

2.2.1. Encoding

Since LDPC codes are linear block codes, it is always possible to encode messages by calculating the generator matrix G from the parity-check matrix H and to multiplicate the messages with G in order to obtain the corresponding codewords. Unfortunately, the generator matrices of LDPC are generally not sparse such that there arise two major drawbacks. The number of multiplications for encoding as well as the memory demand to store the generator matrix is quadratic in the code length. This situation is intensified by the circumstance that we need large code lengths for capacity-achieving decoding performances. As a consequence, it is a big shortcoming of LDPC codes that they usually have a high encoding complexity quadratic in the code length.

However, there exist families of structured LDPC codes whose encoding complexity is linear proportional to the code length. For instance, the class of quasi-cyclic LDPC codes (e.g. [20]) can be encoded by circuits of simple feedback shift registers [21]. One important objective of this thesis is to design structured LDPC codes that are accessible for low-complexity encoding, such that these codes can be used for practical applications.

2.2.2. Iterative decoding

Whereas classical block codes are generally characterized by short code lengths and highly algebraic structure in order to reduce the decoding complexity under ML decoding, LDPC codes pursue a different decoding paradigma. LDPC codes unfold their full strength in combination with *iterative decoding algorithms* which allow the codes to potentially achieve the limit of Shannon's coding theorem. These algorithms were independently discovered several times (e.g. [3, 22]) and thus are known under different names and variants such as the *belief propagation algorithm* [23], the *message passing algorithm* [24] or the *sum-product algorithm* [25].

The family of iterative decoding algorithms can generally be divided into *hard-decision* algorithms which use binary values for their decoding computations and *soft-decision* algorithms which use probabilistic and continuous values. In general, hard-decision algorithms have a low computational complexity, but perform poorly compared to soft-decision algorithms. Conversely, soft-decision algorithms exhibit very good decoding performances but at the cost of a high complexity. As a consequence, there is always a tradeoff between the computational complexity and the error performance of decoding algorithms.

When Gallager invented the LDPC codes in his thesis [4], he also came along with various iterative decoding algorithms, today known as the Gallager A/B algorithms (see, e.g., [26]) and the bit-flipping algorithm. These algorithms are examples for hard-decision decoding and are widely used over the binary symmetric channel (BSC). In the same thesis, Gallager presented the first probabilistic decoding method as an instance of soft-decision algorithms. After it has been recognized that these algorithms perform very well, many new iterative decoding algorithms have been developed for various channels.

The most famous iterative decoding algorithm is the so-called *sum-product algorithm* (SPA). The basic idea behind the SPA is to iteratively improve approximations of the a-posteriori probabilities (APPs) based on the incoming message symbols. For this, the

algorithm sends probabilistic messages iteratively along the edges of the code's factor graph and applies Bayes' rule at each node based on the extrinsic information gained of the local neighboring check nodes. Since the parity-check matrix is sparse, the factor graph has only a small number of edges which makes the decoding algorithm efficient. It can be shown that the SPA leads to correct APPs when the factor graph is free of cycles, but for practical reasons, this assumption can not be met by serious LDPC codes. However, it has been recognized that the SPA nevertheless works amazingly well if the factor graph has cycles. For a detailed description of the SPA, the reader is referred to [27, 28].

The SPA is the best algorithm in terms of the error rate but has also the highest computational complexity among the iterative decoding algorithms. Hence, there exist many modified algorithms that reduce the complexity of the standard SPA at the cost of a degradation in performance, for instance, the *min-sum algorithm* [29]. A survey of belief propagation algorithms with reduced complexity can be found in [30].

2.2.3. Construction principles

The main approaches for the construction of LDPC codes can basically be subdivided into two lines of research.[1] On the one hand, random LDPC codes are constructed by non-deterministic computer algorithms under certain design criteria and, on the other hand, structured LDPC codes are designed based on combinatorial tools such as certain block designs, geometries and finite fields. The primary aim of random-like constructions is to optimize the decoding performance of the arising codes and therefore, random codes generally have an excellent decoding performance, in particular, for large code lengths and small code rates. However, this approach leads to codes that have no inner algebraic structure which can be exploited for more efficient encoding and decoding algorithms and thus are challenging to implement in practice. Moreover, the properties of these codes

[1] Note that this section is a partial reprint of the introductory sections [P1, Section I] and [P2, Section I] which have also been written by the author of this thesis.

can not be guaranteed due to the non-deterministic nature of their random constructions and thus, there is no assurance that a particular code have good properties. As a further consequence, these codes are hard to optimize if some specific properties are required.

The second ongoing line of research focuses on the construction of highly structured LDPC codes based on combinatorial designs. The underlying structure can typically be exploited for low-complexity encoding and an acceleration of the decoding process as opposed to random-like codes. Furthermore, the parity-check matrices can be stored more effectively in a compressed form from which the full parity-check matrices can be calculated on-the-fly. Moreover, structured LDPC codes can guarantee deterministic code properties such as 4-cycle-free factor graphs and are more amenable for extensive mathematical analyses.

A fertile and sophisticated approach is to utilize the well-known concepts of combinatorial design theory for the construction of LDPC codes by considering the incidence matrix of a combinatorial design as the parity-check matrix of an LDPC code. This productive connection of designs and codes has been discovered independently by several researchers (e.g. [31, 32, 33]). Subsequently, structured LDPC codes have been designed based on finite geometries (FGs) [31, 32] and Steiner 2-designs [34, 35, 27, 33], a certain subclass of balanced incomplete block designs (BIBDs). A wide range of structured LDPC codes has also been derived from the field of partial geometries (e.g. [36]), including the codes from Steiner 2-designs as a subclass. Further subclasses are codes on generalized quadrangles as presented in [37], and codes on transversal designs considered in [36, 27]. An important subclass of structured LDPC codes are quasi-cyclic codes (see, e.g., [38] and the references therein). By exploiting their cyclic structure, these codes can be encoding with linear complexity by using circuits of simple feedback shift registers [21, 39].

In order to improve the decoding performance of LDPC codes, there are two usual strategies pursued in the literature. First, by increasing the girth of the code's factor graph (e.g., [40, 41]), i.e., by avoiding the smallest cycles which are known to be harmful

for the iterative decoding process. Large girth speeds up the convergence of iterative decoding and leads to better performance if the number of iterations is limited. Second, by lowering the so-called *error-floors*. This phenomenon is a significant flattening of the *bit-error-rate* (BER) curve beyond a certain *signal-to-noise-ratio* (SNR) and is a problem of focal importance, since many practical applications of LDPC codes require extremely low operational BERs [42]. It has been discovered that error-floors are caused by special substructures in the code's factor graph that act as internal states in which the iterative decoder can be trapped. Richardson [43] introduced the notion of *trapping sets* to describe such internal states for iterative decoders.[2]

Depending on the channel and the iterative decoding algorithm, trapping sets may have quite different characteristics. Over the *binary erasure channel* (BEC), trapping sets have a purely combinatorial representation known as *stopping sets*. These entities completely determine the decoding performance over the BEC [47] and have been studied in a large series of subsequent papers (e.g. [48, 49]). Stopping sets are also extensively investigated from a design theoretic perspective (e.g. [50]), from this viewpoint known as *full configurations* in designs.

For more complex non-erasure channels such as the AWGN channel, trapping sets have a more subtle nature and can not easily be described by a simple combinatorial notion such as stopping sets. However, a major subclass of the occurring trapping sets over the AWGN channel can be described by combinatorial objects called *(fully) absorbing sets* which have been introduced in [51]. It has been demonstrated by extensive hardware simulations [51, 52] that these entities are the main contributors to the error-floors over the AWGN channel under SPA decoding (see Section 2.2.2).

In order to improve the decoding performance in the error-floor region, it is a crucial step to identify those trapping sets of an LDPC code that cause the iterative decoder to fail, and, in a second step, to find techniques to avoid the most harmful ones.

[2] Other publications that are concerned with codes on graphs under iterative decoding have studied closely related notions as elementary trapping sets [44] and pseudocodewords [45, 46].

2.2.4. Stopping sets

Stopping sets are the cause of decoding failures over the BEC under iterative erasure (or peeling) decoding (e.g. [53]) and can be considered as special states in which the decoder gets trapped. It has been shown in [47] that these entities completely determine the decoding performance over the BEC under iterative decoding such that an exact analysis of the bit erasure probability is possible. Basically, a stopping set is a subset of bit nodes of a code's factor graph such that each bit nodes is connected to at least two check nodes. For a more detailed definition of stopping sets, the reader is referred to [48], or, from a design theoretic perspective, to the full version of [50].

In order to assess the quality of an LDPC code over the BEC, it is a substantial step to examine the *stopping set distribution* of the code, meaning the collection of stopping sets that occur in the code's factor graph along with the number of their occurrences [54]. In particular, it is important to identify the smallest stopping sets since they dominate the performance in the error-floor region. The size of the smallest stopping sets is termed the *stopping distance* (or *stopping number*) (e.g., [48, 49]) and plays an essential role under iterative decoding over the BEC comparable to the role of the minimum distance played under maximum-likelihood (ML) decoding.

Knowing the smallest stopping sets of a code is essential for estimating and evaluating the code's performance, even more, since the performance in the error-floor region is usually beyond the scope of classical Monte-Carlo simulations. Unfortunately, it is already an NP-hard problem to find the size of the smallest stopping sets (cf. [55, 42]) such that computer-based search algorithms for finding and enumerating stopping sets reach their limits for growing code lengths unless the code has no specific structure that can be exploited.

It is therefore a reasonable approach to examine code families based on suitable combinatorial designs which provide an algebraic setting for the investigation of stopping sets on a simplified way. These codes have potentially useful structures that can be ex-

ploited to identify and enumerate harmful stopping sets. The challenge is then to reveal those code instances with the most beneficial stopping set distributions, in particular, with the highest stopping distance and the minimum number of dominating stopping sets. This strategy has been pursued in a large number of previous papers (e.g. [50, 54]) and is also the core idea of our publication [P2] as well as of Chapter 4.

2.2.5. Absorbing sets

The notion of *absorbing sets* has been firstly introduced in [51]. By using a hardware emulation platform for investigating the causes of error floors over the AWGN channel, the authors have recognized that the behavior of the SPA performance in the low-BER region can be linked to special substructures of a code's factor graph, termed absorbing sets. For a contemporary definition of absorbing sets in terms of the factor graph, the reader is referred to [13]. Absorbing sets have a purely combinatorial description such as stopping sets over the BEC and can be considered as special subclasses of near-codewords, as proposed in [56], and trapping sets, as introduced in [43]. Also, absorbing sets are closely related to the notion of fixed sets (e.g. [57, 58, 59]) which have been primarily used to characterize the main error events over the BSC when decoding with the Gallager A/B algorithms [4]. In [13], the notion of absorbing sets has been extended to fully absorbing sets which have the property that they are stable under bit-flipping algorithms [60] and thus are the dominant factors for any bit-flipping decoder. Since the decoding failures of various message-passing algorithms and channels are supposed to be closely related [61], it can be assumed that fully absorbing sets in general greatly contribute to the error-floor performance for various iterative decoder and channels.

Since (fully) absorbing sets are the primary cause of decoding failures over the AWGN channel, it is of great importance to identify and count the most harmful absorbing sets that may occur in the factor graph of LDPC codes. Such a classification is extremly useful for various theoretical and practical approaches: it provides a valuable groundwork for the analytical study of error-floors in the low-BER regions, leading to a better theoretical

understanding of the failure mechanisms over the AWGN channel and other related channels. In [62], a deterministic method for predicting the error-floors over different channel models has been developed in terms of absorbing sets and their cardinalities, leading to theoretical bounds for the decoder's performance. Furthermore, in [63], a technique based on importance sampling has been presented in order to estimate the decoding performances in the error-floor region by using the characterization of dominant absorbing sets and their cardinalities. Both methods are even more valuable since they allow the prediction of BERs in regions that are out of reach for standard Monte-Carlo simulations. From a practical viewpoint, the knowlegde of absorbing sets is essential for the design of LDPC codes with improved absorbing set spectra [64, 65], leading to codes with better error-floor performances that can be used for high-speed applications operating in low-BER regions. Also, it greatly supports the design of high-throughput hardware emulators [66, 52].

The identification of absorbing sets in a code's factor graph is known to be a very hard problem such that search algorithms are generally limited to small absorbing sets fulfilling certain constraints which are supposed to have the most detrimental effect on the decoding performance. The identification of such absorbing sets can be significantly enhanced by using the specific structure of LDPC codes based on combinatorial designs which will be demonstrated in the present thesis. The harmfulness of an absorbing set can currently not calculated exactly due to the absence of a complete characterization and understanding of the failure mechanisms over the AWGN channel. However, the harmfulness can be based on several conjectures obtained by intuition and by experimental results. An (a, b) *absorbing set* of size a and syndrome b (cf. [13]) is supposed to be harmful if

(a) the size a is small,

(b) the syndrome b is small compared to a,

(c) the absorbing set is fully, and

(d) the degrees of the bit nodes are small.

2.3. Systematic repeat-accumulate (sRA) codes

A common weakness of many LDPC codes is their high encoding complexity quadratic in the code length, although there are some structural LDPC codes that can be encoded with linear complexity, for instance, quasi-cyclic codes. This handicap motivates the family of systematic repeat-accumulate (sRA) codes which are designed to have an encoding scheme with low complexity. Moreover, the encoding scheme leads to codes that are representable by sparse parity-check matrices such that sRA codes can be decoded in exactly the same powerful way as LDPC codes (see e.g. [10, 67, 68]). Hence, sRA codes are simultaneously a class of fast encodable turbo-like codes and a class of LDPC codes, gaining the advantages of both code representations [10].

2.3.1. Encoding with low complexity

The main idea of sRA codes is a serial concatenation of two simple constituent codes with an interleaver and an optional combiner between them [11]. It has been recognized that these sRA codes, although simple, show a great decoding performance under iterative decoding. In this thesis, we consider only *regular* sRA codes, but they can also easily be generalized to the irregular case by employing an irregular repetition code and optionally an irregular combiner (see [69, 70, 71] and the references therein). The encoding process is precisely described in [72, 11], but for the convenience of the reader, the process is outlined below.

(1) First, we start with a message of K input bits

$$(s_1, s_2, \ldots, s_K).$$

(2) Then, we apply a repetition code with parameter q which means that the i-th bit

will be repeated q times, giving

$$(u_1, u_2, \ldots, u_{qK}) := (\underbrace{s_1, \ldots, s_1}_{\times q}, \underbrace{s_2, \ldots, s_2}_{\times q}, \ldots, \underbrace{s_K, \ldots, s_K}_{\times q}).$$

Clearly, it arises a bit sequence of length qK.

(3) A permutation Π, called an *interleaver*, shuffles the bit sequence and leads to

$$(v_1, v_2, \ldots, v_{qK}) := (\Pi(u_1), \Pi(u_2), \ldots, \Pi(u_{qK})).$$

(4) We apply a *combiner* with parameter a which means that the subsequence will firstly be partitioned into M subsequences such that each subsequence has length a and, secondly, that the bits of each subsequence will be summed up mod 2. Let S_i be the i-th subsequence and define $\oplus(S_i) := \left\{ \sum x \pmod 2 : x \in S_i \right\}$. Then,

$$(r_1, r_2, \ldots, r_M) := (\oplus(S_1), \oplus(S_2), \ldots, \oplus(S_M)).$$

Clearly, it arises a bit sequence of length $M = \frac{qK}{a}$.

(5) Next, an *accumulator* sums up the bit sequence consecutively and outputs

$$\begin{aligned} p_1 &:= r_1, \\ p_i &:= r_i \oplus p_{i-1}, \quad \text{for } i = 2, \ldots, M. \end{aligned}$$

(6) Finally, the K message bits and the M output bits of the accumulator are concatenated, giving the final codeword

$$(s_1, s_2, \ldots, s_K, p_1, p_2, \ldots, p_M)$$

of length $N := K + M$. This is called a *systematic* output.

As previously demonstrated, the encoding scheme of an sRA code can be performed very efficiently due to simple processing steps based on low-level operations. We obtain an sRA code of length $N = K(1 + \frac{q}{a})$ and rate $R = \frac{a}{a+q}$ [70] which is equal to the design rate, since the 2-rank of the parity-check matrix is generally full. Furthermore, the parity-check matrix of an sRA code can always be represented in the form $H = [H_1, H_2]$ with the following properties (cf. [10, 11]):

(1) H_1 is of size $M \times K$ and H_2 of size $M \times M$.

(2) H_1 has column weights q, i.e., the column weights are solely determined by the repetition code.

(3) H_1 has row weights a, i.e., the row weights are solely determined by the combiner.

(4) The structure of H_1 is completely specified by the permutation Π of the interleaver.

(5) H_2 is a double-diagonal matrix with $M-1$ columns of weight two and one column of weight one such that the i-th column has 1-entries at rows i and $i+1$ (whereas the last column has only one 1-entry in the last row).

(6) The structure of H_2 is solely determined by the accumulator.

2.3.2. Decoding as LDPC codes

Systematic RA codes can be considered as a special class of LDPC codes and thus can be decoded on exactly the same way as LDPC codes [10] by using an interative decoding algorithm that operates efficiently on the code's sparse factor graph. This LDPC code representation of sRA codes is clarified by the following theorem.

Theorem 1 ([10]) *Systematic RA codes can be considered as LDPC codes since their parity-check matrices are sparse.*

Proof. This connection will be now elaborated in more detail since it is important for the understanding of the present thesis and since no particular proof has been found in the literature. We start with the canonical basis vectors $\boldsymbol{e_1}, \ldots, \boldsymbol{e_K}$ of $\mathbb{F}_2^K$. From these vectors, we will generate K codewords which form a generator matrix by taking them as rows. With input $\boldsymbol{e_1}$, we have q many 1-entries after the repetition code in step (2). The interleaver in step (3) does not change this number, since it only distributes the 1-entries. We assume that this distribution is designed in such a way that no two points lie in the same subsequence of step (4), since then the interleaver would not be working correctly and no valid parity-check matrix would be arise. Hence, the combiner in step (4) does not cancel out 1-entries through binary additions such that the number of ones

stays the same. The accumulator in step (5) is therefore the only processing step that reduces the number of 1-entries. By writing the output vectors row-wise in a matrix, we obtain the generator matrix of the sRA code in standard form,

$$G = \left[\begin{array}{c:c} & \boldsymbol{p_1} \\ I_K & \vdots \\ & \boldsymbol{p_K} \end{array}\right]$$

where I_K is the $K \times K$ identity matrix due to the systematic output and $\boldsymbol{p_i}$ are the output vectors of the accumulator for the i-th input vector $\boldsymbol{e_i}$. It is well known that if $G = (I_K, X)$ is the generator matrix in standard form, then $H = (-X^T, I_{N-K})$ is the parity-check matrix of the code (e.g., [73]). Therefore, we obtain the parity-check matrix

$$H = \left[\begin{array}{ccc:c} \boldsymbol{p_1^T} & ... & \boldsymbol{p_K^T} & I_M \end{array}\right] = \left[\begin{array}{ccc:cccc} p_{1,1} & ... & p_{K,1} & 1 & 0 & ... & 0 \\ p_{1,2} & ... & p_{K,2} & 0 & 1 & ... & 0 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{1,M} & ... & p_{K,M} & 0 & 0 & ... & 1 \end{array}\right].$$

of the sRA code with $M = N - K$, where the first K columns are the output vectors of the accumulator. The target now is to transform H in such a way that I_M is converted into a double diagonal matrix which is compatible for the sRA encoding scheme. The double diagonal matrix, denoted by H_2, should be of the form

$$H_2 := \left[\begin{array}{ccccc} 1 & & & & \mathbf{0} \\ 1 & 1 & & & \\ & \ddots & \ddots & & \\ & & & & \\ & & 1 & 1 & \\ \mathbf{0} & & & 1 & 1 \end{array}\right].$$

Let $\boldsymbol{z_1}, ..., \boldsymbol{z_M}$ be the rows of H. We modify the rows by

$$\begin{aligned} \boldsymbol{z'_1} &:= \boldsymbol{z_1}, \text{ and} \\ \boldsymbol{z'_i} &:= \boldsymbol{z_i} \oplus \boldsymbol{z_{i-1}}, \text{ for } i = 2, ..., M. \end{aligned}$$

By applying this on the rows of H, we obtain

$$H = \left[\begin{array}{ccc:c} p_{1,1} & \cdots & p_{K,1} & \\ p_{1,2} \oplus p_{1,1} & \cdots & p_{K,2} \oplus p_{K,1} & \\ p_{1,3} \oplus p_{1,2} & \cdots & p_{K,3} \oplus p_{K,2} & H_2 \\ \vdots & & \vdots & \\ p_{1,M} \oplus p_{1,M-1} & \cdots & p_{K,M} \oplus p_{K,M-1} & \end{array}\right].$$

The bit $p_{j,l}$ for $j = 1, ..., K$ and $l = 1, ..., M$ can be replaced by the accumulated sum $p_{j,l} = \sum_{k=1}^{l} r_{j,k}$ where $r_{j,k}$ is the k-th bit of the vector after the combiner with $\boldsymbol{e_j}$ as input vector. It follows that

$$\begin{aligned} p_{j,1} &= r_{j,1}, \\ p_{j,1} \oplus p_{j,2} &= r_{j,1} \oplus \sum_{k=1}^{2} r_{j,k} = r_{j,2}, \\ &\vdots \\ p_{j,K-1} \oplus p_{j,K} &= \sum_{k=1}^{K-1} r_{j,k} \oplus \sum_{k=1}^{K} r_{j,k} = r_{j,K}. \end{aligned}$$

Consequently, the parity-check matrix can be simplified to

$$H = \left[\begin{array}{ccc:c} r_{1,1} & \cdots & r_{K,1} & \\ r_{1,2} & \cdots & r_{K,2} & \\ r_{1,3} & \cdots & r_{K,3} & H_2 \\ \vdots & & \vdots & \\ r_{1,M} & \cdots & r_{K,M} & \end{array}\right] = \left[\begin{array}{ccc:c} \boldsymbol{r_1^T} & \cdots & \boldsymbol{r_K^T} & H_2 \end{array}\right].$$

By transforming the identity matrix to the form of a double diagonal matrix, the calculations of the accumulater are reversed such that the first K columns are the bit vectors $\boldsymbol{r_i}$ after the combiner. As shown above, the vectors $\boldsymbol{r_i}$ have weight q_i. Furthermore, H_2 has $M-1$ columns of weight two and a single column of weight one. By neglecting the weight-1 column, the density ρ_H of the parity-check matrix H can be calculated by

$$\rho_H = \frac{qK + 2M}{NM}.$$

With $K = RN$ (where R is the code rate) and $M = N - RN$, it follows that

$$\rho_H = \frac{qR + 2 - 2R}{N(1 - R)}.$$

With constant rate R and constant q, the density converges against 0 for $N \to \infty$. Therefore, the parity-check matrix H is sparse for large code lengths. ■

2.3.3. Construction principles

The key part of the design of sRA codes with good decoding performances is the establishment of an high-quality interleaver for which there are potentially $(qK)!$ possible permutations. Since sRA codes are specialized LDPC codes, the design of interleavers are implicitly included in the design of a parity-check matrix. Therefore, the typical approach for the construction of a good sRA code is to design a parity-check matrix as in the traditional way for LDPC codes and to adapt it so that it fits into the encoding scheme of an sRA code (cf. [10, 11]). Then, the interleaver for the encoding process can directly be derived from the structure of the parity-check matrix [72]. Since sRA codes are decoded as LDPC codes, they benefit from the same structural properties that are required to design good LDPC codes with respect to their decoding performance, for example, avoiding short cycles in the code's factor graph.

Now, we discuss the process of transferring an LDPC code into an sRA code. Once the parity-check matrix H of an LDPC code has been constructed, for instance, by the combinatorial construction techniques in Section 3, the matrix has to be tranformed into the sRA compatible form $H' = [H_1, H_2]$ as described in Subsection 2.3.1. To keep things easy, we assume that H is free of 4-cycles which is an essential property for an LDPC code with good decoding performances. Also, we assume that the originating LDPC code is regular, since we only consider regular sRA codes. Note that these assumptions

are trivially satisfied by the LDPC codes based on combinatorial designs which will be considered in the present thesis.

(1) First, let $\mathrm{B}(i)$ be the column of H with 1-entries at row i and $i+1 \pmod{M}$ for $1 \leq i \leq M$ where M is the number of rows. It can easily be verified that $\mathrm{B}(i)$ must be unique, since H is free of 4-cycles.

(2) Delete all $\mathrm{B}(i)$ if existent. Then, the remaining columns form H_1.

(3) Finally, append a double diagonal matrix H_2 of size $M \times M$. Note that the columns $\mathrm{B}(i)$ have to be deleted in step (2) in order to avoid 4-cycles in the arising parity-check matrix H'.

(4) Finally, the sRA code parameters q, a, K and Π can be derived from the parity-check matrix H'. The parameters q and a are equal to the constant column and row weight of H_1, respectively, and K is equal to the number of columns of H_1. The only tricky part is the derivation of Π. For this, enumerate the 1-entries of H_1 column-wise and let this number be φ. Furthermore, enumerate the 1-entries of H_1 row-wise and let this number be γ. Now, let $\Pi(s)$ be the the interleaver entry at position s for $1 \leq s \leq qK$. Then, for every 1-entry of H_1 with sequential indices (φ, γ), set $\Pi(\gamma) = \varphi$.

2.4. Guiding principles for good code design

This section gives a short overview about the basic principles for good LDPC code design based on best practice. Furthermore, the presented principles can directly be transferred to the design of sRA codes, since these codes can be considered as special LDPC codes.

2.4.1. Performance vs. ease of implementation

The performance of a code is obviously the most critical factor in code design. Under iterative message passing decoding, the performance of an LDPC code can be broken down into three measurements. First, how good is the ultimate error performance, i.e., how close does the code's performance come to the Shannon limit. Second, how fast does the decoding process converge to its best possible solution and, third, how is the error-floor behavior of the code in the high-SNR region.

Besides having an excellent decoding performance, a good LDPC code should also faciliate an efficient (hardware) implementation for their practical applicability. Unfortunately, the simultaneous fulfilment of both properties is a big challenge, since an easier implementation requires more code structure that can be exploited in order to accelerate the encoding and decoding algorithms, but which also leads to a potential degradation of the decoding performance due to the loss of flexibility in the code design. Hence, there has to be found a convenient trade-off between random-like codes with optimized performances and highly structured codes with an efficient implementation.

In general, randomly constructed LDPC codes have very good decoding performances since they have only few framework conditions for the construction of their factor graphs and thus a maximum degree of freedom for connecting bit nodes with variable nodes. Indeed, the best known LDPC codes are randomly constructed. However, the implementation of such codes is very complex due to the lack of code structure. In particular, there is no general way for encoding random-like LDPC codes with low complexity.

By contrast, structured LDPC codes based on combinatorial designs are subjected to many combinatorial conditions imposed by the underlying designs, but can usually be encoded with linear complexity by exploiting their inner structure. Furthermore, the structure can typically be utilized to accelerate the decoding algorithm and to store the code's parity-check matrix in a storage space saving manner. An important class of structured LDPC codes are *quasi-cyclic LDPC codes* which can be encoded with low complexity by using simple feedback shift registers [21] while offering excellent decoding performances.

2.4.2. Large code length

A large code length is known to be beneficial for the decoding performance since then the parity-check equations typically involve many code bits such that the corresponding check nodes profit from much extrinsic information coming from the neighboring bit nodes. In fact, the best known decoding performance over the AWGN channel has been achieved by an LDPC code with a very large code length of $N = 10^7$ [74]. Although an implementation of this code is rather impratical due to a very high encoding and decoding complexity, it demonstrates that a large code length is highly benefical for the decoding performance of an LDPC code.

2.4.3. Sparsity vs. connectivity

The main idea of an LDPC code is that its parity-check matrix is sparse such that the iterative decoding algorithm operates efficiently on the code's factor graph. Also, a sparse parity-check matrix obviously reduces the memory consumption in order to store the matrix. On the other hand, the degrees of the bit nodes and check nodes should be large enough to gain valuable extrinsic information from the neighboring nodes throughout the decoding process. A usual approach is to keep the column weights relatively small (typically three or slightly larger) which guarantees the sparsity of the parity-check matrix and which has also shown to deliver good decoding results.

2.4.4. Large girth by avoiding short cycles

A major target in the LDPC code design is to increase the girth of the code's factor graph which is defined as the size of the smallest cycle. It is well known that small cycles are harmful for the decoding process since they affect the independence of the extrinsic information exchanged by the iterative decoder [75]. Theoretically, the standard SPA converges to the optimal solution as long as the factor graph is free of cycles (e.g. [76]), but nonetheless, the design of pratical LDPC codes with reasonable rates inevitably leads to cycles in their factor graphs. By applying the SPA on these codes, the existence of small cycles decelerate the speed of convergence which finally leads to a degraded decoding performance when the number of iterations is limited.[3] Conversely, large girth speeds up the convergence of iterative decoding and leads to a better performance if the number of iterations is limited. Hence, the focus of many papers that design structured codes lies on achieving a large girth (e.g., [40, 41]), but at the cost of their code rates.

On the other hand, it has been shown that the SPA applied on factor graphs with relatively small cycles can nevertheless result in astonishingly good decoding performances since these cycles can be compensated by a well designed code structure. Many papers that are concerned with high-rate structured LDPC codes are typically content with avoiding the most harmful 4-cycles in order to maintain the high code rates. The cycles of size 6 and higher then can be compensated by the relatively long code lengths and inner structure of the resulting codes.

2.4.5. Full rank vs. rank deficient parity-check matrices

Adding linearly dependent rows to a code's parity-check matrix can be beneficial for the decoding performance over various channels (e.g. [48, 77, 78]). While the code rate

[3] From a theoretical perspective, the existence of cycles prevents an exact error-probability analysis of the iterative decoding and for smaller cycles, the analysis breaks down earlier [33].

is unaffected, the additional rows lead to more parity-check equations such that each code bit is checked by more parity-check equations. This finally results in an improved decoding performance which has been demonstrated, for example, in [27] and [32], where families of LDPC codes based on rank deficient parity-check matrices have been designed.

However, the gain of performance is small compared to the increase of the decoding effort, such that this approach is not reasonable for high-speed applications where the time efficiency of the decoding algorithm is a crucial factor. For such applications, it is recommendable to use a full ranked parity-check matrix since it minimizes the decoding effort for a given code rate while having a reasonable decoding performance. In the present thesis, we mainly consider LDPC codes whose highly combinatorial constructions lead to full or nearly full ranked parity-check matrices in order to design codes for high-speed applications such as magnetic recording and optical communications channels.

2.4.6. Increasing the minimum distance

The minimum distance $d_{min}(\mathcal{C})$ of the code should be maximized. This strategy is primarily important for ML decoding, whereas the failure mechanisms for iterative decoding is more subtle. However, since every codeword defines a stopping set and an absorbing set in the code's factor graph, the codewords with the smallest weights also lead to small and extremly harmful stopping sets and to the smallest possible absorbing sets of syndrome 0. Hence, an increase of the minimum distance is also highly beneficial for iterative decoding over various channels, in particular, over the BEC and the AWGN channel.

2.4.7. Low error-floors

For high-speed applications that operate at very high SNRs, an essential requirement to the error-correcting code is that error-floors are lowered significantly. These error-floors are caused by trapping sets such as stopping sets over the BEC (Subsection 2.2.4) or

absorbing sets over the AWGN channel (Subsection 2.2.5). As a consequence, it is very important to avoid these substructures in the code's factor graph in order to improve the decoding performance in the critical low error-floor region. Since obviously only a small fraction of trapping sets can be avoided, it is reasonable to focus on the most harmful ones which are typically small and, in the case of absorbing sets, have small syndromes.

2.4.8. Summary

As a summary, it can be observed that the design of LDPC and sRA codes has to cope with many trade-offs and is therefore strongly dependent on the specific application for which a code is used. In the present thesis, the main objective is to design structured LDPC codes for high-speed applications such as magnetic recording or optical communication channels. Such codes must typically have high code rates already at small to moderate code lengths and low error-floors in order to operate successfully in high SNR regions. The low error-floors are achieved by avoiding the most relevant stopping and absorbing sets in the code's factor graph. General requirements are that the parity-check matrices are sparse, (nearly) full-ranked and free of harmful 4-cycles. Very importantly, the designed codes must have an efficient (hardware) implementation.

3

New Combinatorial Constructions for LDPC and sRA Codes

This chapter presents novel and infinite classes of structured LDPC codes and systematic RA codes based on combinatorial designs, more precisely, on balanced incomplete block designs (BIBDs). The arising codes are characterized by very high code rates already at small to moderate block lengths and by their specific structure that can be exploited for highly efficient encoding and decoding algorithms. Hence, the codes are suitable for high-speed applications such as magnetic recording or communication channels. Moreover, the factor graphs of these codes exhibit good structural properties with regard to their decoding performances, in particular, they have no harmful cycles of size four.

3.1. Preliminaries

This section gives some standard material on combinatorial designs which will be important for the current chapter. For encyclopedic references, the reader is referred to [79, 80]. Note that this section is basically a reprint of [P1, Section II].

3.1.1. Balanced incomplete block designs (BIBDs)

Let $\mathcal{P}$ be a set of v elements and $\mathcal{B}$ a collection of k-subsets of $\mathcal{P}$. The elements of $\mathcal{P}$ and $\mathcal{B}$ are called *points* and *blocks*, respectively. An ordered pair $(\mathcal{P}, \mathcal{B})$ is defined to be a *balanced incomplete block design*, denoted by BIBD(v, k, λ), if each pair of points is contained in exactly λ blocks. A BIBD$(v, k, 1)$ is also called a *Steiner 2-design*. It is straightforward that in a BIBD each point is contained in the same number r of blocks, and for the total number b of blocks, the parameters of a BIBD satisfy the relations $bk = vr$ and $\lambda(v-1) = r(k-1)$. For $(\mathcal{P}, \mathcal{B})$ a BIBD(v, k, λ), its *incidence matrix* is a $v \times b$ matrix, denoted by $\mathcal{N}$, in which $\mathcal{N}_{ij} = 1$ if the i-th point of $\mathcal{P}$ is contained in the j-th block of $\mathcal{B}$, and $\mathcal{N}_{ij} = 0$ otherwise. Clearly, $\mathcal{N}$ is unique up to column and row permutation.

Let $(\mathcal{P}, \mathcal{B})$ be a BIBD(v, k, λ), and let σ be a permutation on $\mathcal{P}$. For a block $B = \{b_1, \ldots, b_k\} \in \mathcal{B}$, define $B^\sigma := \{b_1^\sigma, \ldots, b_k^\sigma\}$. If $\mathcal{B}^\sigma := \{B^\sigma : B \in \mathcal{B}\} = \mathcal{B}$, then σ is called an *automorphism* of $(\mathcal{P}, \mathcal{B})$. If there exists an automorphism σ of order v, then the BIBD is called *cyclic* and is denoted by CBIBD(v, k, λ). In this case, the point set $\mathcal{P}$ can be identified with $\mathbb{Z}_v$, the set of integers modulo v, and σ can be represented by $\sigma : i \to i + 1 \pmod{v}$. For a block $B = \{b_1, \ldots, b_k\}$ in a CBIBD(v, k, λ), the set $B + i := \{b_1 + i \pmod{v}, \ldots, b_k + i \pmod{v}\}$ for $i \in \mathbb{Z}_v$ is called a *translate* of B, and the set of all distinct translates of B is called the *orbit* containing B. If the length of an orbit is v, then the orbit is said to be *full*, otherwise *short*. A block chosen arbitrarily from an orbit is called a *base block* (or *starter block*). If k divides v, then the orbit containing the block $B = \left\{0, \frac{v}{k}, 2\frac{v}{k}, \ldots, (k-1)\frac{v}{k}\right\}$ is called a *regular short orbit*. For a CBIBD$(v, k, 1)$ to exist, a necessary condition is $v \equiv 1$ or $k \pmod{k(k-1)}$. When $v \equiv 1 \pmod{k(k-1)}$ all orbits are full, whereas if $v \equiv k \pmod{k(k-1)}$ one orbit is the regular short orbit and the remaining orbits are full.

A BIBD is said to be *resolvable*, and denoted by RBIBD(v, k, λ), if the block-set $\mathcal{B}$ can be partitioned into classes $\mathcal{R}_1, \ldots, \mathcal{R}_r$ such that every point of $\mathcal{P}$ is contained in

exactly one block of each class. The classes $\mathcal{R}_i$ are called *resolution* (or *parallel*) *classes*. If $\mathcal{R}_i$ is a resolution class, define $\mathcal{R}_i^\sigma := \{B^\sigma : B \in \mathcal{R}_i\}$. An RBIBD is called *cyclically resolvable* if it has a non-trivial automorphism σ of order v that preserves its resolution $\{\mathcal{R}_1, \ldots \mathcal{R}_r\}$, i.e., $\{\mathcal{R}_1^\sigma, \ldots \mathcal{R}_r^\sigma\} = \{\mathcal{R}_1, \ldots \mathcal{R}_r\}$ holds. If, in addition, the design is cyclic with respect to the same automorphism σ, then it is called *cyclically resolvable cyclic*, and denoted by CRCBIBD(v, k, λ).

In a CBIBD$(v, k, 1)$, we can define a multiset $\Delta B := \{b_i - b_j : i, j = 1, \ldots, k; i \neq j\}$ of *differences* for a base block $B = \{b_1, \ldots, b_k\}$. Let $\{B_i\}_{i \in I}$, for some index set I, be all the base blocks of full orbits. If $v \equiv 1 \pmod{k(k-1)}$, then clearly $\bigcup_{i \in I} \Delta B_i = \mathbb{Z}_v \setminus \{0\}$. The family of base blocks $\{B_i\}_{i \in I}$ is then called a *cyclic difference family* in $\mathbb{Z}_v$, denoted by CDF$(v, k, 1)$.

Let k be an odd positive integer and $p \equiv 1 \pmod{k(k-1)}$ a prime. A CDF$(p, k, 1)$ is said to be *radical*, and denoted by RDF$(p, k, 1)$, if each base block is a coset of the k-th roots of unity in $\mathbb{Z}_p$ [81]. If there exists a RDF$(p, k, 1)$ with k odd, then there exists a CRCBIBD$(kp, k, 1)$ [82].

3.1.2. LDPC codes based on BIBDs

This section summarizes the state-of-the-art of general LDPC code design based on BIBDs (cf. [32, 33, 34, 35, 27]): The $v \times b$ incidence matrix of a BIBD$(v, k, 1)$ can be considered as a parity-check matrix H of a binary $(k, r = \frac{bk}{v})$-regular *BIBD LDPC code* of length $N = b = \frac{v(v-1)}{k(k-1)}$, if we identify the v points with the parity-check equations and the b blocks with the N code bits. In this case, all column weights of H are equal to k, and all row weights are equal to r. The dimension of the code is $K = b - \mathrm{rank}_2(H)$, where $\mathrm{rank}_2(H)$ denotes the 2-rank of H, and the code rate is $R = 1 - \mathrm{rank}_2(H)\frac{k(k-1)}{v(v-1)}$. In general, the calculation of $\mathrm{rank}_2(H)$ depends on the specific structure of the BIBD$(v, k, 1)$. It is clear from the classical Fisher's Inequality that $\mathrm{rank}_2(H) \leq v$. Furthermore, it is elementary to see that, if $2 \mid \frac{v-1}{k-1}$, then $\mathrm{rank}_2(H) \geq v - 1$, with equality if and on-

ly if $2 \mid k$ (cf. [83, Theorem 2.4.1]). More precise results are in particular known for BIBD$(v, 3, 1)$s [84].

As in a BIBD$(v, k, 1)$ no pair of points have more than one block in common (i.e., no pair of columns of H contains more than one "1" at the same positions), the Tanner graph of a BIBD-LDPC code is free of cycles of length four which are known to be very harmful for the decoding process. Moreover, since every pair of points occurs in exactly one block, it follows that the girth is exactly six. Since each column in H has weight k and no two columns have more than one "1" in common, there are k parity-check equations that are orthogonal on every code bit. Thus, the minimum distance of the code is at least $k + 1$. The same lower bound holds for the size of minimal stopping sets [50].

Some specific classes of BIBDs proved to be especially useful in the code design:

(a) LDPC code constructions based on CDFs, called *CDF LDPC codes*, have been considered in [34, 35, 27] for constant column weights $k = 3$ to 7. Since CBIBDs can easily be constructed from CDFs (e.g. [35, Section III]), these codes form a subclass of the more general class of LDPC codes based on CBIBDs, called *CBIBD LDPC codes*. In the following, we distinguish between these two classes since CDF LDPC codes are in general quasi-cyclic [85] and thus can be encoded with linear complexity, whereas there are some CBIBD LDPC codes that does not have this property. More specifically, all CBIBD LDPC codes based on a CBIBD with a regular short orbit are not quasi-cyclic by definition, although these codes have a highly cyclic structure.

(b) LDPC codes based on RBIBDs, called *RBIBD LDPC codes*, have been constructed in [27, 33] for constant column weights $k = 3, 4$.

(c) LDPC codes based on CRCBIBDs, called *CRCBIBD LDPC codes*, have been designed in [27, 33] for constant column weights $k = 3, 4$.

3.2. New high-rate LDPC codes based on BIBDs

In the first part of our paper [P1, Section III], we have proposed some infinite families of LDPC codes based on several subclasses of BIBDs, more specifically, on CBIBDs, RBIBDs and CRCBIBDs. By using the incidence matrices of these BIBDs as the parity-check matrices of LDPC codes we obtain structured code families with deterministic properties. This idea is not new and has been taken up in a various number of previous manuscripts (e.g. [32, 34, 35, 27, 33]). However, the results of our paper are more general then previous ones. The presented codes are 4-cycle-free and cover a wide spectrum of column weights and code lengths while offering very high rates. It can easily be shown that these codes have the highest code rates among all regular 4-cycle-free LDPC codes of the same column weight, the same number of parity-check equations and the same or higher rank of their parity-check matrices [27]. This statement is formalized in the following proposition.

Proposition 2 *Let H be the $N \times M$ parity-check matrix of a BIBD LDPC code with column weight k and rate R, and let H' be the $N' \times M$ parity-check matrix of any regular 4-cycle-free LDPC code of the same column weight and rate R' such that $\text{rank}_2(H') \geq \text{rank}_2(H)$. Then, it follows that $R \geq R'$.*

Proof. For BIBDs, every pair of points occurs exactly once by definition. There are $\frac{M(M-1)}{2}$ possible pairs of points and every block contains $\frac{k(k-1)}{2}$ pairs. Hence, it follows that the code length of the BIBD LDPC code is given by $N = \frac{M(M-1)}{k(k-1)}$. For any regular 4-cycle-free LDPC code, every pair of points is contained in at most one block (when considering the columns as blocks and the rows as points), leading to $N' \leq \frac{M(M-1)}{k(k-1)}$. It follows directly that $N' \leq N$ and hence $R' \leq R$ using that $\text{rank}_2(H') \geq \text{rank}_2(H)$. ■

3.2.1. CBIBD LDPC codes

Thm. 3 proposes some infinite classes of CBIBD LDPC codes with column weights ranging from 3 to 7 and two finite classes with column weights of 8 and 9. The CBIBDs are based on known families of cyclic difference families (CDFs) in [79, 86] and the references therein and thus are, more specifically, CDF LDPC codes. It should be noted that subclasses of these codes have already been considered in [34, 35, 27].

Theorem 3 [P1, Thm. 1] *Let p be a prime. Then there exists a regular quasi-cyclic CDF LDPC code of column weight k, length $N = \frac{p(p-1)}{k(k-1)}$ and rate $R \geq 1 - \frac{k(k-1)}{v-1}$ based on a CDF$(p, k, 1)$ for the following cases:*

(1) $(k, p) = (3, 6t + 1)$ *for any positive integer t,*

(2) $(k, p) = (4, 12t + 1)$ *for any positive integer t such that v is a prime power,*

(3) $(k, p) = (5, 20t + 1)$ *for any positive integer t such that v is a prime power,*

(4) $(k, p) = (6, 30t + 1)$ *for any positive integer $t \neq 2$,*

(5) $(k, p) = (7, 42t + 1)$ *for any positive integer $t > 1$ with the possible exceptions $p = 127, 211$ as well as primes $p \in [261239791, 1.236597 \times 10^{13}]$ such that $(-3)^{\frac{q-1}{14}} = 1$ in $\mathbb{Z}_p$.*

(6) $(k, p) = (8, p)$ *for all values of $p \equiv 1 \pmod{56} < 10^4$ with the possible exceptions $p = 113, 169, 281, 337$,*

(7) $(k, p) = (9, p)$ *for all values of $p \equiv 1 \pmod{72} < 10^4$ with the possible exceptions $p = 289, 361$.*

The spectrum of the new code families is partially visualized in Fig. 3.1. As we can observe, almost all CDF LDPC codes achieve a code rate of at least 0.9. More precisely, the codes of the smallest column weight three exceed the bound $R \geq 0.9$ already at a small code length of approximately 600 and the codes of the highest column weight nine at a code length of roughly 10^4. Hence, the presented codes achieve very high rates along with manageable small to moderate code lengths.

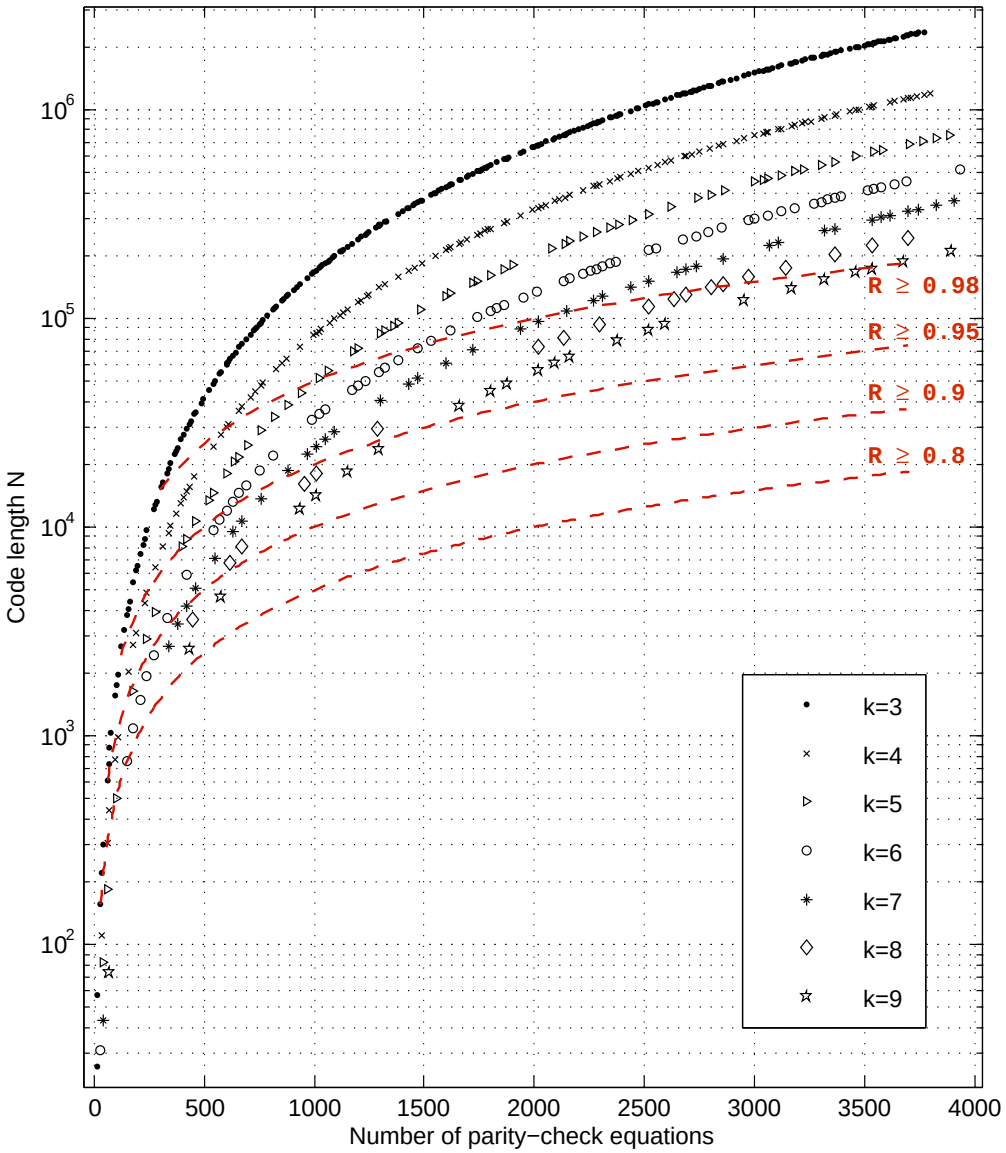


Abbildung 3.1. – The black markers represent possible CDF LDPC codes, each based on a CDF$(p, k, 1)$ with k varying from 3 to 9. The plot relates the number of parity-check equations (equivalent to the prime order p of the CDF) to the code length $N = p(p-1)/k(k-1)$ where k is equal to the column weight of the code. The plot also visualizes some upper bounds for which the codes definitely achieve a rate of $R \geq 0.8, 0.9, 0.95$ and 0.98.

An essential strength of CDF LDPC codes is the quasi-cyclic structure of their parity-check matrices consisting of circulant submatrices that correspond to the block orbits of the underlying CBIBDs [P1]. The benefit of this structure is twofold. Firstly, quasi-cyclic codes can be encoded with circuits of simple feedback shift register [21], leading to a low encoding complexity linear in the block length. Secondly, by removing an arbitrary number of circulant submatrices, we can adjust the rate and length of the codes more independently, gaining much more flexibility in the code design. This motivates the following theorem.

Theorem 4 *For any $(k, r = \frac{M-1}{k-1})$-regular CDF LDPC code with M parity-check equations, length $\frac{rM}{k}$ and rate of at least $\frac{r-k}{r}$, there exists a wide range of regular 4-cycle-free LDPC codes of column weight k, row weight $r-kt$, code length $\frac{rM}{k-tM}$ and rate of at least $\frac{r-kt-k}{r-kt}$ for $t = 1, \ldots, \frac{r}{k} - 2$.*

Proof. The parity-check matrix of a CDF LDPC code consists of $\frac{r}{k}$ circulant submatrices of size $M \times M$ which correspond to the full orbits of the underlying CBIBD. Clearly, these circulant submatrices are (k, k)-regular. By removing t arbitrarily chosen circulants, a new LDPC code with the specified parameters arises. Note that we have to maintain at least two circulants (i.e., $t \leq \frac{r}{k} - 2$), otherwise the parity-check matrix would be quadratic which prevents reasonable data transmission under the assumption that the parity-check matrix has a full or nearly full 2-rank. ■

Thm. 4 does not specify which of the circulants should be removed. Obviously, there may arise LDPC codes with quite different decoding performances depending on which combination of circulants has been discarded. Therefore, it is an interesting approach to identify the best selection of circulants for a given pair of channel and decoding algorithm. It is unknown if this issue can be solved analytically, but reasonable choices for a given scenario could certainly be found by employing computer simulations.

Tabelle 3.1. – Possible exceptions: An RBIBD$(v, k, 1)$ with $k = 5$ or 8 is not known to exist for the following values of $v \equiv k \pmod{k(k-1)}$, see [P1] © *2012 IEEE.*

	$k = 5$			
v	45	345	465	645

	$k = 8$							
	176	624	736	1128	1240	1296	1408	1464
	1520	1576	1744	2136	2416	2640	2920	2976
	3256	3312	3424	3760	3872	4264	4432	5216
	5720	5776	6224	6280	6448	6896	6952	7008
v	7456	7512	7792	7848	8016	9752	10200	10704
	10760	10928	11040	11152	11376	11656	11712	11824
	11936	12216	12328	12496	12552	12720	12832	12888
	13000	13280	13616	13840	13896	14008	14176	14232
	21904	24480						

3.2.2. RBIBD LDPC codes

Thm. 5 proposes some infinite series of regular RBIBD LDPC codes with column weights varying from 3 to 8 and one infinite class of regular RBIBD LDPC codes for any prime power k, all admitting very high code rates with manageable block lengths. These codes rely on known families of RBIBDs (cf. [79, 87] and the references therein) and it is noted that the codes of column weight 3 and 4 have already been considered in [27, 33].

Theorem 5 [P1, Thm. 2, 3] *Let v be a positive integer. Based on an RBIBD$(v, k, 1)$, there exists a $(k, \frac{v-1}{k-1})$-regular RBIBD LDPC code of length $N = \frac{v(v-1)}{k(k-1)}$ and rate $R \geq 1 - \frac{k(k-1)}{v-1}$ for the following cases:*

(1) $(k, v) = (3, 6t + 3)$ *for any positive integer t,*

(2) $(k, v) = (4, 12t + 4)$ *for any positive integer t,*

(3) $(k, v) = (5, 20t + 5)$ *for any positive integer t with the possible exceptions given in Table 3.1,*

(4) $(k, v) = (8, 56t + 8)$ *for any positive integer t with the possible exceptions given in Table 3.1.*

(5) If v and k are both powers of the same prime, there exists a code with the specified parameters if and only if $(v-1) \equiv 0 \pmod{(k-1)}$ and $v \equiv 0 \pmod{k}$.

These results are visualized in Fig. 3.2 for codes up to roughly 7000 parity-check equations. As we can observe, the proposed families comprise a large spectrum of RBIBD LDPC codes with column weights varying from 3 to 27 and with code rates ranging from 0.6 to nearly 1, where almost all codes have a code rate of at least 0.9. Moreoever, the code families of column weights 3, 4, 5 and 8 exists for a high density of parameters and their code rates exceed the extremly high upper code rate bound $R \geq 0.98$ for manageable code lengths of about $10^{4.2}, 10^{4.5}, 10^{4.7}$ and $10^{5.1}$, respectively.

An important strength of RBIBD LDPC codes is the specific structure of their parity-check matrices consisting of portions of $(k, 1)$-regular submatrices which correspond to the resolution classes of the underlying RBIBDs. By removing an arbitrary number of such resolution submatrices, we can adjust the rate and length of the code more independently, gaining much more flexibility in the code design. This technique is not new and has already been proposed in [33]. Applied on our results, we can construct further regular LDPC codes with parameters given by the following theorem.

Theorem 6 *Given any $(k, r = \frac{M-1}{k-1})$-regular RBIBD LDPC code with M parity-check equations, length $\frac{rM}{k}$ and rate of at least $\frac{r-k}{r}$, there exists a wide range of regular 4-cycle-free LDPC codes of column weight k, row weight $r-t$, code length $\frac{(r-t)M}{k}$ and a rate of at least $\frac{r-k-t}{r}$ for $t = 1, \ldots, r-k-1$.*

Proof. The parity-check matrix of an RBIBD LDPC code consists of r resolution submatrices of size $(M \times \frac{M}{k})$ which are $(k, 1)$-regular. By removing a selection of t arbitrary resolutions, we produce an LDPC code with the specified parameters. Note that we have to maintain at least $k+1$ resolutions (i.e., $t \leq r-k-1$), otherwise no data transmission is possible under the assumption of a full or nearly full 2-rank. ■

It is worth noting that the resolution submatrices of RBIBD LDPC codes have only $\frac{M}{k}$ columns, whereas the circulant submatrices of CBIBD LDPC codes are quadratic and

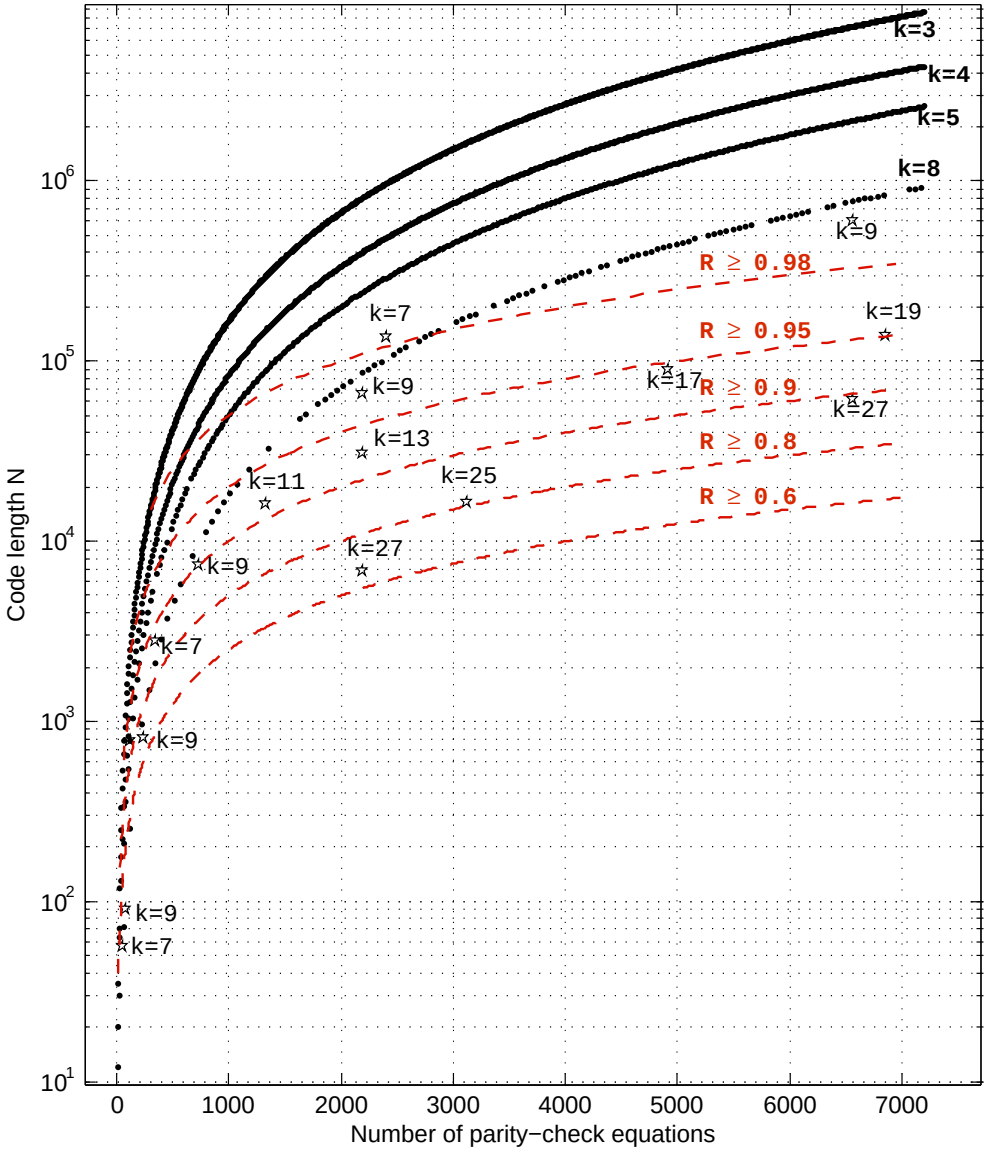


Abbildung 3.2. – The black dots represent possible RBIBD LDPC codes, each based on a RBIBD$(v, k, 1)$ with k varying from 3 to 27. The plot relates the number of parity-check equations (equivalent to the order v of the RBIBD) to the code length $N = v(v-1)/k(k-1)$ where k is equal to the column weight of the code. The plot also visualizes some upper bounds for which the codes definitely achieve a rate of $R \geq 0.6, 0.8, 0.9, 0.95$ and 0.98. Note that all codes with a rate of smaller than 0.1 have been discarded, since these codes have no pratical relevance.

Tabelle 3.2. – Existence of a CRCBIBD$(pk, k, 1)$ with $k = 5$, $p < 10^3$, and $k = 7$ or 9, $p < 10^4$, see [P1] © *2012 IEEE.*

	$k = 5$								
p	41	61	241	281	401	421	601	641	661
	701	761	821	881					

	$k = 7$								
	337	421	463	883	1723	3067	3319	3823	3907
p	4621	4957	5167	5419	5881	6133	8233	8527	8821
	9619	9787	9829						

	$k = 9$						
p	73	1153	1873	2017	6481	7489	7561

have M columns. Hence, RBIBD LDPC codes can be adjusted more smoothly compared to CBIBD LDPC codes and thus allow a more flexible code design. In this context, it is an interesting question which selection of resolution submatrices lead to the best decoding performance in terms of a given channel and decoding algorithm, but the problem will not be treated in the present thesis.

3.2.3. CRCBIBD LDPC codes

Thm. 7 proposes some infinite series of regular CRCBIBD LDPC codes with column weights varying from 3 to 9 and with moderate to very high code rates. These code classes are based on known families of radical difference families (RDFs) (cf. [82, 81, 79, 88, 89] and the references therein) and have partially been treated in [27, 33].

Theorem 7 [P1, Thm. 4] *Let p be a prime number. Based on a CRCBIBD$(pk, k, 1)$, there exists a $(k, \frac{pk-1}{k-1})$-regular CRCBIBD LDPC code of length $N = \frac{p(pk-1)}{k-1}$ and rate $R \geq 1 - \frac{k(k-1)}{pk-1}$ for the following cases:*

(1) $(k, p) = (3, 6t + 1)$ for any positive integer t,

(2) $(k, p) = (4, 12t + 1)$ for any odd positive integer t,

(3a) $(k, p) = (5, 20t + 1)$ *for any positive integer* t *such that* $p < 10^3$*, and furthermore*

(3b) $(k, p) = (5, 20t + 1)$ *for any positive integer* t *satisfying the following condition:* $\varepsilon + 1$ *is not a* 2^{e+1}*-th power in* $\mathbb{Z}_p$*, or equivalently* $(11 + 5\sqrt{5})/2$ *is not a* 2^{e+1}*-th power in* $\mathbb{Z}_p$*, where* 2^e *is the largest power of* 2 *dividing* t *and* ε *a* 5*-th primitive root of unity in* $\mathbb{Z}_p$*,*

(4) $(k, p) = (7, 42t + 1)$ *for any positive integer* t *satisfying the following condition: there exists an integer* f *such that* 3^f *divides* t *and* $\varepsilon+1, \varepsilon^2+\varepsilon+1, \frac{\varepsilon^2+\varepsilon+1}{\varepsilon+1}$ *are* 3^f*-th powers but not* 3^{f+1}*-th powers in* $\mathbb{Z}_p$*, where* ε *is a* 7*-th primitive root of unity in* $\mathbb{Z}_p$*,*

(5) $(k, p) = (9, p)$ *for the values of* $p \equiv 1 \pmod{72} < 10^4$ *given in Table 3.2.*

Moreover, based on a CRCBIBD$(\ell k, k, 1)$*, there exists a* $(k, \frac{\ell k-1}{k-1})$*-regular CRCBIBD LDPC code of length* $N = \frac{\ell(\ell k-1)}{k-1}$ *and rate* $R \geq 1 - \frac{k(k-1)}{\ell k-1}$ *for the following cases:*

(6) (k, ℓ) *for* $k = 3, 5, 7,$ *or* 9*, and* ℓ *is a product of primes of the form* $p \equiv 1 \pmod{k(k-1)}$ *as in the cases above,*

(7) $(k, \ell) = (4, \ell)$ *and* ℓ *is a product of primes of the form* $p = 12t + 1$ *with* t *odd.*

The results are partially visualized in Fig. 3.3. We can observe that there is a high density of CRCBIBD LDPC codes for column weights 3 to 5 with code rates of 0.9 and higher. Moreover, in the observed range up to about 3700 parity-check equations, there exist three CRCBIBD LDPC codes of column weight $k = 7$ and code rates of at least 0.98, and one code of column weight $k = 9$ with a code rate of nearly 0.9.

The major advantage of CRCBIBD LDPC codes is that they inherit the resolvability of RBIBD LDPC codes and the cyclic character of CBIBD LDPC codes such that they combine the specific benefits of both code classes [P1]. More specifically, these codes allow low-complexity encoding by exploiting their cyclic structure and have the same parametric flexibility as codes based on RBIBDs by removing an arbitrary number of small resolution classes. In particular, since CRCBIBDs are also CBIBDs and RBIBDs, the results of Thm. 4 and 6 can be transferred to CRCBIBD LDPC codes, such that we

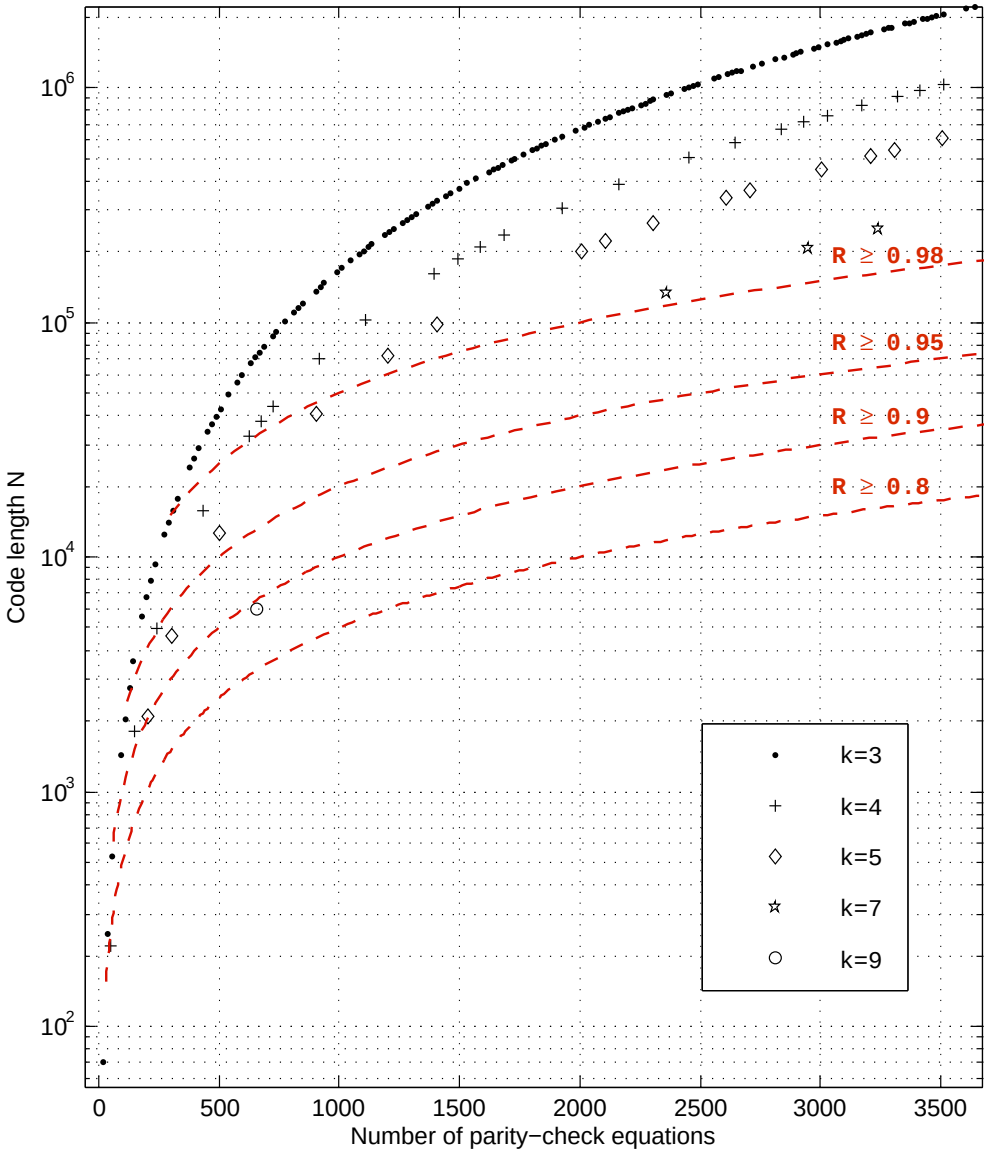


Abbildung 3.3. – The black markers represent possible CRCBIBD LDPC codes, each based on a CRCBIBD$(v, k, 1)$ with k varying from 3 to 9. The plot relates the number of parity-check equations (equivalent to the order v of the CRCBIBD) to the code length $N = v(v-1)/k(k-1)$ where k is equal to the column weight of the code. The plot also visualizes some upper bounds for which the codes definitely achieve a rate of $R \geq 0.8, 0.9, 0.95$ and 0.98.

obtain a wide range of regular LDPC codes where the rates and lengths can be chosen more independently.

3.3. New high-rate sRA codes based on BIBDs

Based on known families of BIBDs, this section presents several new classes of sRA codes and generalized weight-q sRA codes with combinatorial interleavers. Note that these codes have already been considered in the second part of our paper [P1, Section VI]. Depending on the underlying type of BIBD, the arising codes are called CBIBD, RBIBD and CRCBIBD sRA codes, respectively.

3.3.1. Weight-q sRA codes

A drawback of the classical sRA code is that the H_2-part of its parity-check matrix has $M-1$ columns of weight two and one column of weight one which result in a low error-floor in the code's performance. More precisely, by considering the code's factor graph, the columns of weight two correspond to bit nodes of degree two and are a weakness in the decoding process since these bit nodes gain only little extrinsic information from the two neighboring check nodes. However, in the case of sRA codes with very large code lengths, the influence of these deficient columns is small since the code length is quadratic proportional to M while we have only M columns of lower weight. As a consequence, the decoding performance converges against the performance of the corresponding LDPC codes, but along with the low-complexity encoding of sRA codes.

For the case of short to moderate code lengths, Johnson and Weller developed a refined encoding scheme [90, 10] by proposing an alternative accumulator structure that reduces the number of weight-two columns significantly. The resulting codes are termed *weight-3 RA* (w3RA) codes, since the accumulator produces mainly columns of weight three in the H_2-part of their parity-check matrices.

Inspired by the work of Johnson and Weller, we have extended their results by proposing a generalized accumulator design in [P1, Section VI] which leads to novel codes termed *weight-q sRA* codes. It is noted that this modified accumulator has been similarly introduced by Liva et. al [91], but with focus on the design of irregular RA codes and flexible choices of the node degree distributions. The new accumulator structure enables weight-q sRA codes for higher column weights without suffering the low error-floors as in the case of classical sRA codes.

Recall the encoding scheme in Section 2.3.1. We now replace the accumulator of step (5) by a new generalized accumulator. For this, let $g_1, \ldots, g_{q-1} \in \{1, ..., M\}$ be the design parameters of the generalized accumulator and $s_\ell := \sum_{j=1}^{\ell} g_j$ for $\ell = 1, \ldots, q-1$. Furthermore, let r_i denote the output of the combiner at time i. Then, the output p_i of the new accumulator is calculated by the recursive scheme

$$p_i \;\; := \;\; r_i \oplus p_{i-s_1} \oplus ... \oplus p_{i-s_{q-1}}, \quad \text{for } i = 1, ..., M$$

where $p_{i-s_\ell} = 0$ if $i - s_\ell < 0$. This accumulator again results in a parity-check matrix of the form $H = [H_1, H_2]$, where the i-th column of H_2 has 1-entries in the rows i and $i + s_\ell$, $1 \leq \ell \leq q-1$, given that these rows exist. Thus, the parameters g_j, called the *accumulator design parameters*, specify the vertical distance between the j-th and $(j+1)$-th 1-entry in the columns of H_2. The H_1-part is unaffected by the accumulator and hence remains as introduced in Section 2.3.1. Consequently, a large fraction of the H_2-columns have the same high column weight as the columns of H_1, leading to an improved decoding performance compared to sRA codes.

The new accumulator produces $(M - s_{q-1})$ columns of weight q. Since high column weights are benefical for the decoding process, the number of weight-q columns should be maximized. This means that the accumulator design parameters have to be as small as possible in order to minimize $s_{q-1} = g_1 + \ldots + g_{k-1}$. On the other hand, it must hold that $g_i \neq g_j$ for $i \neq j$ in order to avoid 4-cycles in the code's factor graph. The best choice for the accumulator design paramaters is therefore $g_i = i$ for $i = 1, \ldots, q-1$.

Based on this generalized accumulator, we use the combinatorial construction techniques presented in Section 3.2 to construct new families of (weight-q) sRA codes based on CBIBDs, RBIBDs and CRCBIBDs (cf. [P1, Section VI]). The new code classes have an excellent decoding performance very close to those of regular LDPC codes, along with the low encoding complexity of turbo-like codes. In the following, the particular code classes will be described in more detail.

3.3.2. CBIBD sRA codes

The specific structure of a CBIBD$(v, k, 1)$ can generally be exploited for the construction of a weight-q sRA code of column weight k for any $2 \leq q \leq k$ (where with $q = 2$ the classical sRA code arises).

Theorem 8 *Let $(\mathcal{P}, \mathcal{B})$ be the points and blocks of a CBIBD$(v, k, 1)$ where the points $\mathcal{P}$ are identified with $\mathbb{Z}_v$. Let p_1 and p_2 be any two points of $\mathcal{P}$ such that $\gcd(p_2 - p_1, v) = 1$ and let $B = \{p_1, p_2, \ldots, p_k\}$ be the block of $\mathcal{B}$ containing these points. Then, there exists a CBIBD weight-q sRA code for any $2 \leq q \leq k$ of length $N = \frac{v(v-1)}{k(k-1)}$, rate $R = 1 - \frac{k(k-1)}{v-1}$ and with accumulator design parameters*

$$g_i = (p_2 - p_1)^{-1}(p_{i+1} - p_i) \pmod{v}, \text{ for } i = 1, \ldots, q-1.$$

Clearly, $g_1 = 1$. The remaining design parameters $g_2, \ldots, g_{q-1}$ depend on the base block which contains the chosen pair of points. In order to minimize the g_i values, all possible combinations of points should be tested for each base block.

Proof. Proof by construction: define a permutation σ on $\mathcal{P}$ by $\sigma : x \to (p_2 - p_1)^{-1}(x - p_1) \pmod{v}$ for any point $x \in \mathbb{Z}_v$. Next, construct the incidence matrix of $(\mathcal{P}^\sigma, \mathcal{B}^\sigma)$ such that the point $x \in \mathbb{Z}_v$ is identified with row $x + 1$. Let $C_i = (c_1, \ldots, c_v) \in \{0, 1\}^v$ be the column where $c_i = 1$ and $c_{i+1} = 1$ for $i = 1, \ldots, v$. Then, set $c_j = 0$ if $j < i$ or $j > \min\{\ell : \sum_{l=i}^{\ell} c_l \geq q\}$. Now, define $H_2 = [C_1, \ldots, C_v]$ and take the remaining columns as H_1. Finally, the parity-check matrix $H = [H_1, H_2]$ defines a CBIBD weight-q sRA code with the specified parameters. ■

This process has been demonstrated in [P1, Subsections VI-A] based on some CDF constructions known in the literature. Note that it is straightforward to obtain a CBIBD once a CDF is known (e.g. [35, Section III]). Since our aim ist to maximize the harmful low-weight columns in the code's parity-check matrix, we only treat the case when $q = k$. In the following, a summarization of our results is given:

(1) From Netto's "first" construction [92], we obtain a CDF$(v, 3, 1)$ for prime powers of the form $v \equiv 1 \pmod 6$ and thus also a CBIBD$(v, 3, 1)$. With Thm. 8, we can construct a series of CBIBD w3RA codes of column weight three, length $N = \frac{v(v-1)}{6}$, rate $R = \frac{v-7}{v-1}$, $M = v$ parity-check equations and accumulator design parameters $g_1 = 1$ and g_2, where g_2 is determined by the CDF. Table 3.3 lists the parameters of the first 22 CBIBD w3RA codes based on Netto's first construction, restricting v to primes. The values of the row (*) are measurements for the success of our w3RA codes (in percent), meaning that this percentage of columns in the accumulator matrix H_2 has been increased compared to the classical sRA counterparts. As we can observe, all listed codes have over 50 percent of increased columns in H_2 and seven of them even over 80 percent. Hence, all codes highly benefit from their w3RA code representation. The values of the row (**) are measurements for the influence of the deficient columns of column weights one and two with respect to the code length. As we can see, the influence of the deficient columns decreases for larger block lengths and thus can be neglected for sufficient large block lengths. Fig. A.1 depicts the parity-check matrices of a CBIBD LDPC, sRA and w3RA code of length $N = 57$ based on an CBIBD$(19, 3, 1)$.

(2) From the construction of Buratti [93], we obtain a series of CDF$(p, 4, 1)$ for primes of the form $p \equiv 1 \pmod{12}$ and a family of CDF$(p, 5, 1)$ for primes of the form $p \equiv 1 \pmod{20}$. The first series leads to CBIBD w4RA codes of column weight four, length $N = \frac{p(p-1)}{12}$, rate $R = \frac{p-13}{p-1}$, $M = p$ parity-check equations and accumulator design parameters $g_1 = 1, g_2$ and g_3 where the values of g_2 and g_3 can not be chosen freely, but are determined by the CBIBD. Table 3.4 lists the parameters of the first 12 w4RA codes up to a code length of 8138 obtained by Buratti's

construction. It can be seen, that nearly all codes have more than 90 percent of columns in H_2 with increased column weights and thus highly profit from their w4RA code representation. Furthermore, the percentage of deficient columns in H rapidly decreases for a growing code length and thus are less influential. The second series leads to CBIBD w5RA codes of column weight five, length $N = \frac{p(p-1)}{20}$, rate $R = \frac{p-21}{p-1}$, $M = p$ parity-check equations and accumulator design parameters $g_1 = 1, g_2, g_3$ and g_4 where the values of g_2, g_3 and g_4 are determined by the CBIBD. In Table 3.5, the parameters of the first 10 w5RA codes based on Buratti's construction are listed. As we can see, all codes have increased column weights for over 90 percent of their H_2-columns and thus highly benefit from their w5RA code representation. Also, the percentage of deficient columns is very small with respect to the code length and thus are expected to have only little influence on the code's performance.

3.3.3. RBIBD sRA codes

A weakness of RBIBD LDPC codes from Section 3.2.2 is that they are not encodable by feedback shift registers as opposed to CDF LDPC codes due to their missing quasi-cyclic structure. However, the specific structure of certain RBIBDs can be utilized to employ an sRA encoder in order to enable turbo-like encoding with linear complexity. Johnson and Weller demonstrated this approach [10] by constructing RBIBD sRA codes based on known RBIBD$(v, 3, 1)$s from Ray-Chaudhuri and Wilson [94]. In our paper [P1, Subsection IV-B], we have modified this construction in order to design novel RBIBD w3RA codes. Table 3.6 lists the first 12 codes where the order v are restricted to be primes of the form $v \equiv 1 \pmod 6$. Note that such codes can also easily be constructed for prime powers. Fig. A.2 visualizes and compares the parity-check matrices of RBIBD LDPC, sRA and w3RA codes of length $N = 70$ based on an RBIBD$(21, 3, 1)$ from Ray-Chaudhuri and Wilson.

Tabelle 3.3. – Parameters of a series of CBIBD w3RA codes of column weight three up to length 8251 based on Netto's "first" construction. Every column represents a code that relies on a CDF$(p, 3, 1)$ with p prime of the form $p \equiv 1 \pmod 6$. The code has length $N = \frac{p(p-1)}{6}$, $M = p$ parity-check equations, rate $R = \frac{p-7}{p-1}$ and accumulator design parameters $g_1 = 1$ and g_2. The codes have been constructed by Thm. 8 in such a way that the design parameters g_i are minimized.

p	13	19	31	37	43	61	67	73	79	97	103
N	26	57	155	222	301	610	737	876	1027	1552	1751
R	0.5	0.67	0.8	0.83	0.86	0.9	0.91	0.92	0.92	0.94	0.94
g_2	3	7	5	10	6	13	29	8	23	35	46
$(*)$	69	58	81	70	84	77	55	88	70	63	54
$(**)$	15.4	14.0	3.9	5.0	2.3	2.3	4.1	1.0	2.3	2.3	2.7

p	109	127	139	151	157	163	181	193	199	211	223
N	1962	2667	3197	3775	4082	4401	5430	6176	6567	7385	8251
R	0.94	0.95	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.97	0.97
g_2	45	19	42	32	12	58	48	84	92	14	39
$(*)$	58	84	69	78	92	64	73	56	53	93	82
$(**)$	2.3	0.7	1.3	0.9	0.3	1.3	0.9	1.4	1.4	0.2	0.5

$*$ Percentage of columns with increased weights in H_2, i.e., $100\frac{M-g_1-g_2}{M}$
$**$ Percentage of deficient columns of weight 1 or 2 in H, i.e., $100\frac{g_1+g_2}{N}$

3.3.4. CRCBIBD sRA codes

The task of encoding codes based on CRCBIBDs is more subtle compared to codes on CDFs. Although CRCBIBDs have a cyclic structure, their LDPC codes are not quasi-cyclic by definition [27], since every CRCBIBD possesses a regular short orbit which leads to an "incomplete" circulant submatrix in the code's parity-check matrix. As a consequence, fast encoding can either be realized by a modified circuit of feedback shift registers as proposed in [27] or, even simpler, by employing the encoder of an sRA code.

In [P1, Subsection IV-C], we have constructed new CRCBIBD weight-q sRA codes based on CRCBIBDs from the direct construction of Genma, Mishima and Jimbo [82]. For this, we have utilized a certain subset of k resolution classes in order to construct

Tabelle 3.4. – Parameters of CBIBD w4RA codes of column weight four up to code length 8138 based on Buratti's construction. Every column represents a code that relies on a CDF$(p, 4, 1)$ with p prime of the form $p \equiv 1 \pmod{12}$. The code has length $N = \frac{p(p-1)}{12}$, $M = p$ parity-check equations, rate $R = \frac{p-13}{p-1}$ and accumulator design parameters $g_1 = 1$, g_2 and g_3. The codes have been constructed by Thm. 8 in such a way that the design parameters g_i are minimized.

p	37	61	73	97	109	157	181	193	229	241	277	313
N	111	305	438	776	981	2041	2715	3088	4351	4820	6371	8138
R	0.67	0.8	0.83	0.88	0.89	0.92	0.93	0.94	0.95	0.95	0.96	0.96
g_2	2	4	6	3	2	8	2	5	29	14	38	13
g_3	21	6	34	55	57	26	30	30	130	210	97	182
$(*)$	92	92	90	96	97	94	98	97	87	94	86	96
$(**)$	2.7	1.6	1.6	0.5	0.3	0.4	0.1	0.2	0.7	0.3	0.6	0.2

$*$ Percentage of columns with increased weights in H_2, i.e., $100\frac{M-g_1-g_2}{M}$
$**$ Percentage of deficient columns of weight 1 or 2 in H, i.e., $100\frac{g_1+g_2}{N}$

Tabelle 3.5. – Parameters of CBIBD w5RA codes of column weight five up to length 38764 based on Buratti's construction. Every column represents a code that relies on a CDF$(p, 5, 1)$ with p prime of the form $p \equiv 1 \pmod{20}$. The code has length $N = \frac{p(p-1)}{20}$, $M = p$ parity-check equations, rate $R = \frac{p-21}{p-1}$ and accumulator design parameters $g_1 = 1, g_2, g_3$ and g_4. The codes have been constructed by Thm. 8 in such a way that the design parameters g_i are minimized.

p	401	421	461	601	641	661	701	761	821	881
N	8020	8841	10603	18030	20512	21813	24535	28918	33661	38764
R	0.95	0.95	0.96	0.97	0.97	0.97	0.97	0.97	0.98	0.98
g_2	5	4	11	21	53	39	9	11	12	19
g_3	48	35	70	122	120	281	277	93	523	181
g_4	284	160	188	356	69	28	236	243	64	236
$(*)$	99	99	97	96	92	94	99	98	98	98
$(**)$	0.07	0.06	0.11	0.12	0.26	0.18	0.04	0.04	0.04	0.05

$*$ Percentage of columns with increased weights in H_2, i.e., $100\frac{M-g_1-g_2}{M}$
$**$ Percentage of deficient columns of weight 1 or 2 in H, i.e., $100\frac{g_1+g_2}{N}$

Tabelle 3.6. – Parameters of RBIBD w3RA codes of column weight $k = 3$ and $N \leq 15862$ based on the RBIBD construction of Ray-Chaudhuri und Wilson [94]. Every column represents a code that relies on a RBIBD$(p, 3, 1)$ with p restricted to primes of the form $p \equiv 1 \pmod 6$. The code has length $N = \frac{p(p-1)}{6}$, $M = p$ parity-check equations, rate $R = \frac{p-7}{p-1}$ and accumulator design parameters $g_1 = 1$ and g_2. The codes have been constructed by Thm. 8 in such a way that the design parameters g_i are minimized.

p	21	39	57	93	111	129	183	201	219	237	291	309
N	70	247	532	1426	2035	2752	5551	6700	7957	9322	14065	15862
R	0.7	0.84	0.89	0.93	0.95	0.95	0.97	0.97	0.97	0.97	0.98	0.98
g_2	16	22	7	67	10	49	13	163	154	181	61	226
$(*)$	19	41	86	27	90	61	92	18	29	23	79	27
$(**)$	24.3	9.3	1.5	4.8	0.5	1.8	0.3	2.4	1.9	2.0	0.4	1.4

$*$ Percentage of columns with increased weights in H_2, i.e., $100\frac{M-g_1-g_2}{M}$
$**$ Percentage of deficient columns of weight 1 or 2 in H, i.e., $100\frac{g_1+g_2}{N}$

the accumulator matrix H_2. These k resolution classes are cyclic shifts of each other and form a single block orbit, whereas all other resolution classes consist of blocks that are distributed over several block orbits. Hence, in order to maintain the resolvable structure, the accumulator matrix H_2 should be constructed from the single orbit that consists of only k resolution classes. The resolved structure can then be used for optimization, for example, by removing a certain selection of the remaining resolution classes. A disadvantage of this approach is that we are restricted to accumulator design parameters that are predefined by the structure of the single block orbit.

On the other hand, since CRCBIBDs are cyclic, we can apply the technique presented in Thm. 8 in order to obtain weight-q RA codes with the best possible accumulator design parameters, but which may lead to a split of the resolution classes. Table 3.7 lists the basic parameters of some CRCBIBD w3RA codes with the best possible accumulator design parameters. Fig. A.3 (c) depicts the parity-check matrix of a CRCBIBD$(21, 3, 1)$ w3RA code.

Tabelle 3.7. – Parameters of CRCBIBD w3RA codes of column weight $k = 3$ up to length 9322 based on CRCBIBDs from the construction of Genma, Mishima and Jimbo [82]. Every column represents a code that relies on a CRCBIBD$(3p, 3, 1)$ with p prime of the form $p \equiv 1 \pmod 6$. The code has length $N = \frac{p(3p-1)}{2}$, $M = pk$ parity-check equations, rate $R = 1 - \frac{6}{3p-1}$ and accumulator design parameters $g_1 = 1$ and g_2. The codes have been constructed by Thm. 8 in such a way that the design parameters g_1 and g_2 are minimized.

p	7	13	19	31	37	43	61	67	73	79
N	70	247	532	1426	2035	2752	5551	6700	7957	9322
M	21	39	57	93	111	129	183	201	219	237
R	0.70	0.84	0.89	0.93	0.95	0.95	0.97	0.97	0.97	0.97
g_2	4	16	7	25	10	49	13	37	64	55
$(*)$	76	56	86	72	90	61	92	81	70	76
$(**)$	7.1	6.9	1.5	1.8	0.5	1.8	0.3	0.6	0.8	0.6

$*$ Percentage of columns with increased weights in H_2, i.e., $100\frac{M-g_1-g_2}{M}$
$**$ Percentage of deficient columns of weight 1 or 2 in H, i.e., $100\frac{g_1+g_2}{N}$

3.4. Simulations

This section serves to demonstrate the decoding power of the novel classes of structured LDPC codes and systematic RA codes by extensive Monte-Carlo simulations. These simulations are performed over the AWGN channel by employing the standard sum-product algorithm [28] with a maximum of 50 iterations per codeword. The results are visualized in Fig. 3.4, 3.5 and 3.6. A legend displays the following information in the respective order: code type, construction method in brackets, and a triple $[N, R, k]$, where N is the code length, R the code rate and k the constant column weight of the parity-check matrix. The structured BIBD LDPC, sRA and weight-q sRA codes are compared to random LDPC codes based on the progressive-edge growth (PEG) algorithm [95]. It can generally be observed that the structured codes outperform their randomly constructed counterparts significantly. Hence, these codes are working very well, in particular, for short to moderate code length and high code rates. The simulations also demonstrate that the new BIBD weight-q sRA codes outperform their classical sRA counterparts

and behave similar as the corresponding LDPC codes. Hence, by employing the novel accumulator structure, we can avoid the high error-floors that occur for classical sRA codes caused by the low weight columns in their parity-check matrices.

Further experimental results for our new combinatorial code families are presented in [P1, Subsection III-D] and in [P1, Subsection IV-D]. In this paper, we have show that the structured codes outperform random regular Gallager LDPC codes [4] with comparable parameter sets. Also, we have observed a performance gain as the column weights grow larger and a particular good performance of CRCBIBD LDPC codes

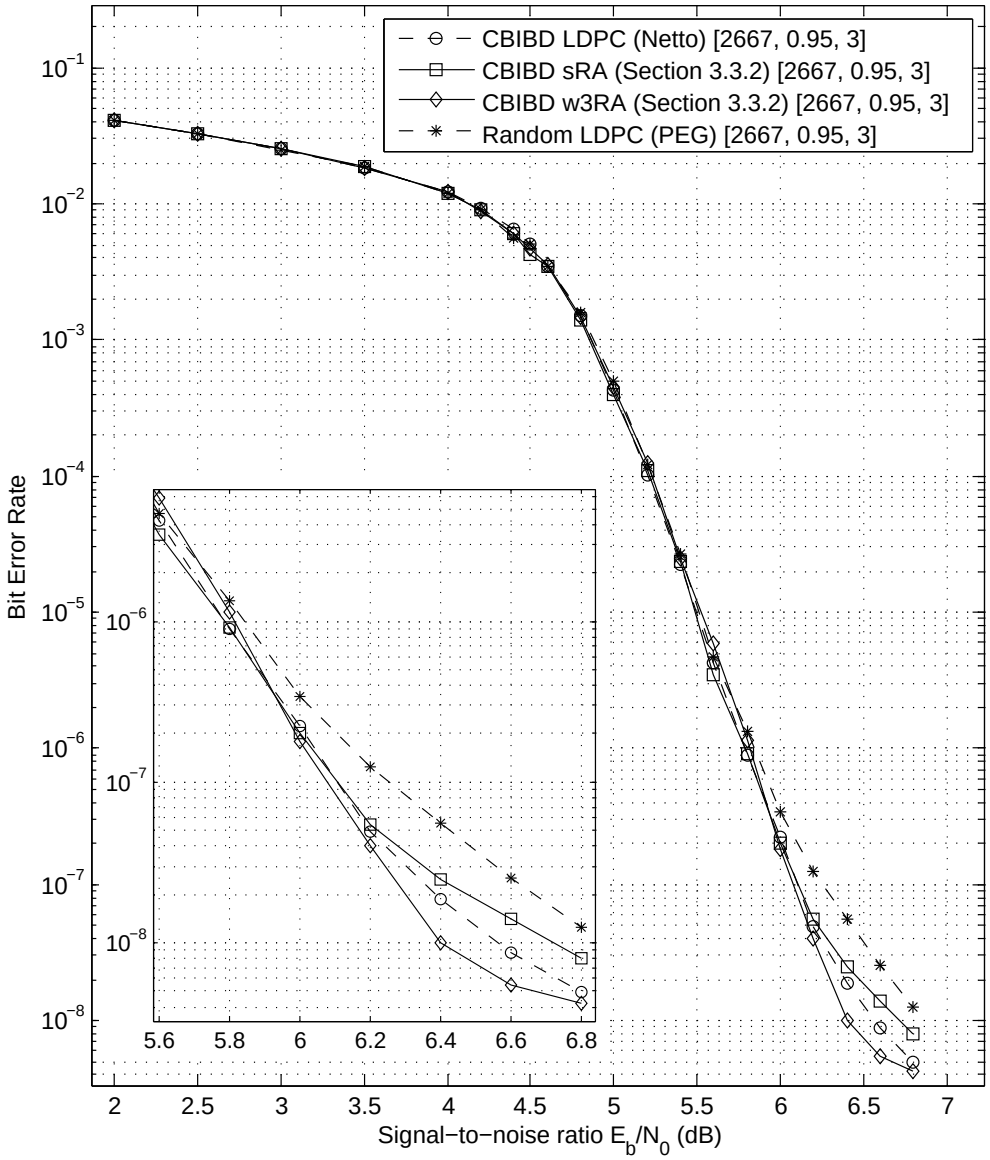


Abbildung 3.4. – The figure compares the decoding performances of a CBIBD LDPC, sRA and w3RA code based on Netto's "first" construction [35] with a random LDPC code constructed by the PEG algorithm. All codes have the same code length $N = 2667$, rate $R = 0.95$ and column weight $k = 3$. It can be observed that the structured BIBD codes outperform their random counterpart significantly at an SNR region of 5.8 dB and larger. Furthermore, the CBIBD sRA code shows a relatively high error-floor already at 5.8 dB, while the corresponding CBIBD w3RA even outperforms the LDPC code.

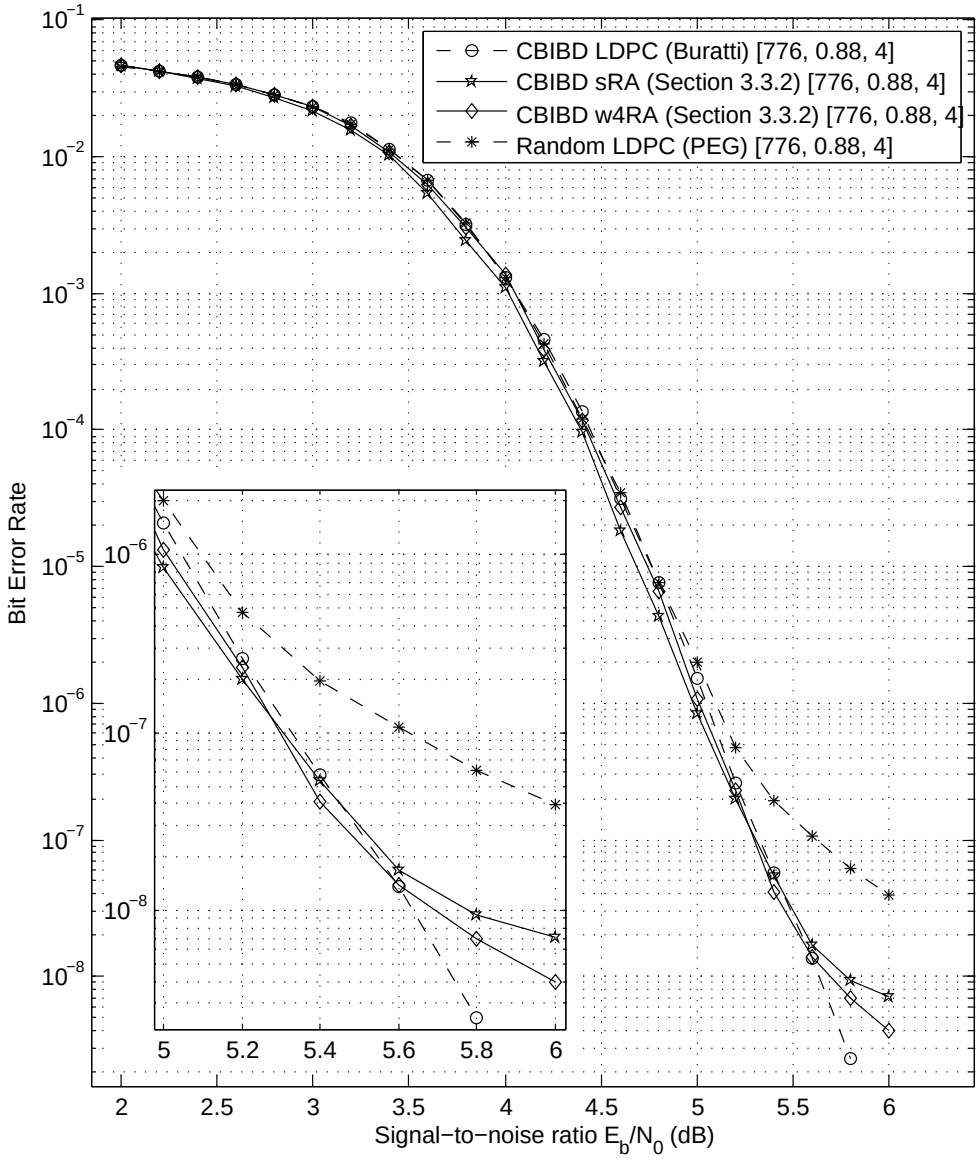


Abbildung 3.5. – The figure compares the decoding performances of a CBIBD LDPC, sRA and w3RA code based on Buratti's construction [93] with a random LDPC code of the same parameters constructed by the PEG algorithm. The structured LDPC code have originally been proposed by Vasic and Milenkovic [35], whereas the sRA and w3RA code are presented in Section 3.3.2. All codes have the same code length $N = 776$, rate $R = 0.88$ and column weight $k = 4$. It can be observed that the structured BIBD codes outperform their randomly constructed counterpart already at an SNR region of 5 dB and larger. While the CBIBD sRA code shows a relatively high error-floor at approximately 5.5 dB, the corresponding CBIBD w3RA lowers this error-floor significantly. Since the code length is relatively small, the impact of the low column weights of H_2 is strong such that both RA codes suffer an error-floor compared to their LDPC counterpart.

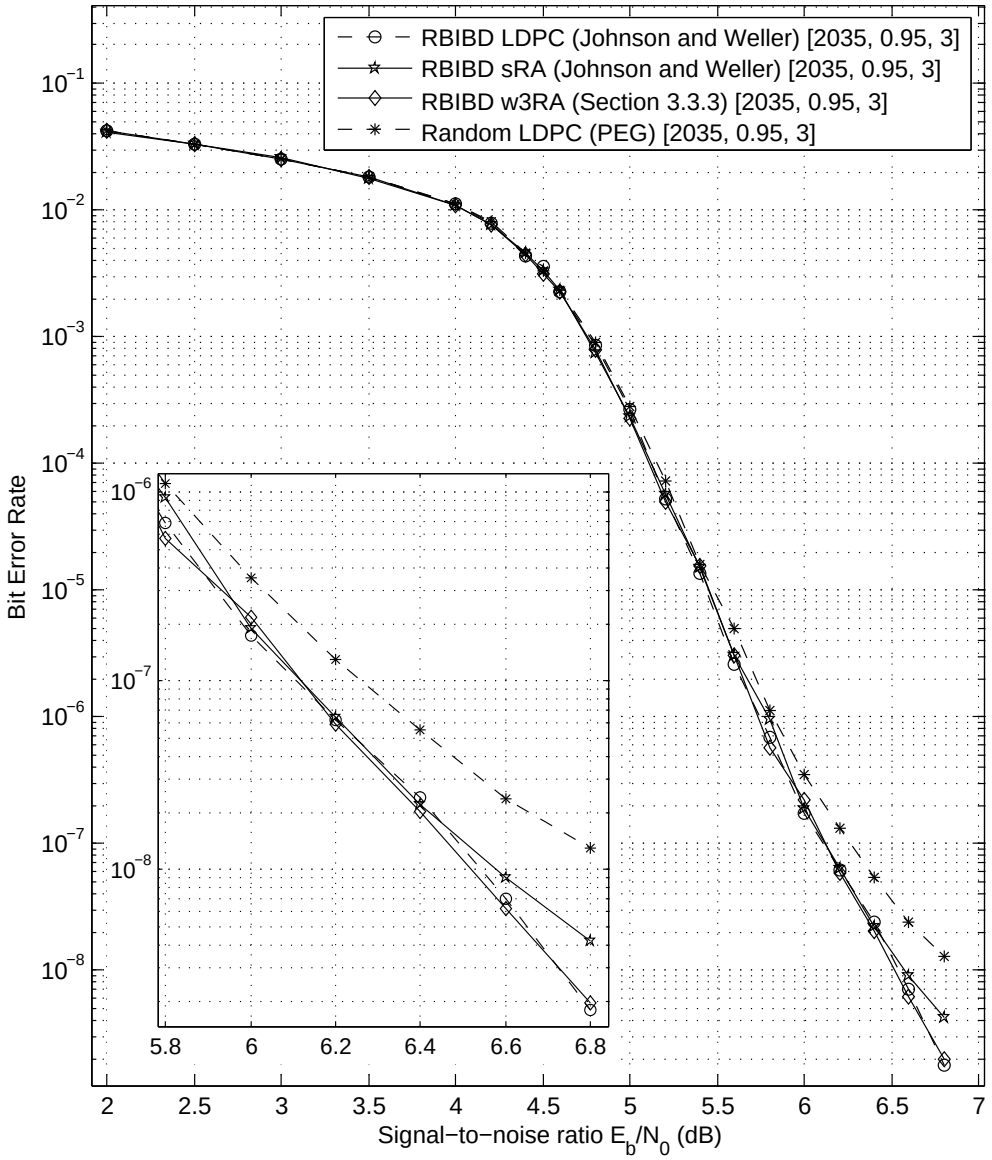


Abbildung 3.6. – The figure shows the decoding performances of an RBIBD LDPC, sRA and w3RA code based on a family of RBIBDs constructed by Ray-Chaudhuri and Wilson [94]. The LDPC and sRA codes have originally been proposed by Johnson and Weller in [33] and [10], respectively, whereas the w3RA code is presented in Section 3.3.3. The structured codes are compared with a random LDPC code based on the PEG algorithm. All codes have the same code length $N = 2035$, rate $R = 0.95$ and column weight $k = 3$. It can be observed that the structured BIBD codes outperform their randomly constructed counterpart significantly already at an SNR of 5.5 dB and larger. Furthermore, the CBIBD sRA code shows a relatively high error-floor at 6.4 dB, while the corresponding CBIBD w3RA avoids this error-floor and behaves similar to the LDPC code.

4

LDPC Codes Based on Transversal Designs

This chapter is concerned with the design of an infinite family of structured LDPC codes based on transversal designs, called *TD LDPC codes*. These codes are thoroughly investigated and designed in order to optimize their decoding performances over the BEC and AWGN channel. The presented results are based on our closely related papers [P2] and [P3], but extend their outcome in several ways. Note that the examined codes have already been considered in [36, 27], but in a rather general way.

After defining some basic concepts in Section 4.1, a special class of cyclic-structured TD LDPC is introduced in Section 4.2. Based on this code class, Section 4.3 provides a thorough stopping set analysis as a foundation for the design of optimized TD LDPC codes over the BEC in Section 4.4. Similarly, in Section 4.5, an extensive absorbing set analysis is performed as a basis for Section 4.6, which treats the design of TD LDPC codes over the AWGN channel.

As a result, there arise infinite families of regular high-rate TD LDPC codes with excellent decoding performances and very low error-floors over the BEC and AWGN channel. When designed properly, these codes are quasi-cyclic and can be encoded with linear complexity by circuits of feedback shift registers. Furthermore, the codes guarantee

the same structural properties than BIBD LDPC codes, in particular, their factor graphs are free of 4-cycles, having a girth of six.

Moreover, transversal designs have the important property that they are resolvable [36], meaning that their blocks can be partitioned into resolution classes such that each point of the design is contained in exactly one block per class. As already discussed in Section 3 for the case of RBIBD LDPC codes, the resolvability of the underlying design of block size k leads to portions of $(k, 1)$-regular submatrices in the code's parity-check matrix. By discarding an arbitrary selection of such matrices, a wide range of regular LDPC codes with various rates and lengths can be derived. This technique has already been applied to codes from Euclidean geometry in [96] and Steiner 2-designs in [97].

4.1. Preliminaries

This section is basically a reprint of [P3, Section II] and gives an overview of the basic concepts that are required to construct LDPC codes based on transversal designs.

4.1.1. Set systems

A *set system*, denoted by $\mathcal{S}$, is a pair $(\mathcal{P}, \mathcal{B})$ consisting of a point set $\mathcal{P}$ and a block set $\mathcal{B}$ which is a family of subsets of $\mathcal{P}$. For a consistent representation, we only allow set systems where $\mathcal{P} = \bigcup \mathcal{B}$ such that each point is contained in at least one block. The *size* of $\mathcal{S}$ is given by the number of blocks of $\mathcal{B}$. The *degree* of a point is the number of blocks containing the point and by $O(\mathcal{P})$ we mean the subset of points of $\mathcal{P}$ having odd degree. Two set systems $(\mathcal{P}, \mathcal{B})$ and $(\mathcal{P}', \mathcal{B}')$ are *isomorphic*, if there exists a bijection between $\mathcal{P}$ and $\mathcal{P}'$ that maps $\mathcal{B}$ to $\mathcal{B}'$. Given a transversal design $\mathcal{D}$ with points $\mathcal{P}$ and blocks $\mathcal{B}$, a set system $(\mathcal{P}, \mathcal{B})$ is called a *configuration* of $\mathcal{D}$ if $\mathcal{P} \subseteq \mathcal{P}$ and $\mathcal{B} \subseteq \mathcal{B}$.[1] Every set system can be equivalently described by a matrix which allows a convenient representation of

[1] The notion of *configurations* in designs is frequently used in the literature about combinatorial designs and codes from designs (e.g. [50]).

the system. More precisely, a set system is equivalent to a $|\mathcal{P}| \times |\mathcal{B}|$ matrix where the entry at (i, j) is one if the i-th point of $\mathcal{P}$ is in the j-th block of $\mathcal{B}$, else zero.

Let $\mathcal{S} = (\mathcal{P}, \mathcal{B})$ be a set system of block size k and Q be a set of k colors. Then, a *k-coloring* of $\mathcal{S}$ is a mapping $\varphi : \mathcal{P} \to Q$ such that the points of any block of $\mathcal{B}$ are colored uniquely, i.e., $|\{\varphi(x) : x \in B\}| = k$ for any $B \in \mathcal{B}$. For each $c \in Q$, the set $\varphi^{-1}(c) = \{x \in \mathcal{P} : \varphi(x) = c\}$ is called a *color class*. Clearly, the union of the color classes of any k-coloring φ define a partition of the points of $\mathcal{P}$. In the present chapter, such partitions are used to describe the k-colorings of a set system. A set system is called *k-colorable* if there is at least one k-coloring. Clearly, there can be different k-colorings for a single set system.

Let $\mathcal{S} = (\mathcal{P}, \mathcal{B})$ and $\mathcal{S}' = (\mathcal{P}', \mathcal{B}')$ be two set systems of block size k with k-colorings φ and φ', respectively. Then,

(1) $\mathcal{S}$ is isomorphic to $\mathcal{S}'$ if and only if there exists a bijection $\sigma : \mathcal{P} \to \mathcal{P}'$ such that $\sigma(\mathcal{B}) = \mathcal{B}'$ where $\sigma(\mathcal{B}) := \big\{\{\sigma(x) : x \in B\} : B \in \mathcal{B}\big\}$.

(2) $\mathcal{S}_\varphi$ is isomorphic to $\mathcal{S}'_{\varphi'}$ if and only if, in addition to (1), there exists a bijection $\phi : Q \to Q$ such that $\phi(\varphi(x)) = \varphi'(\sigma(x))$ for any $x \in \mathcal{P}$.

4.1.2. Latin squares

A *Latin square* L of order n is an array of $n \times n$ cells, where each row and each column contains every symbol of an n-set S exactly once [79]. Let $L[x, y]$ denote the symbol at row $x \in X$ and column $y \in Y$, where X and Y are n-sets indexing the rows and columns of L, respectively. Two Latin squares L_1 and L_2 of order n are *orthogonal*, if they share a common row and column set X and Y, respectively, and if the ordered pairs $(L_1[x, y], L_2[x, y])$ are unique for all $(x, y) \in X \times Y$. In other words, there can not be two cell positions $[x_1, y_1]$ and $[x_2, y_2]$ such that $L_1[x_1, y_1] = L_1[x_2, y_2]$ and $L_2[x_1, y_1] = L_2[x_2, y_2]$. A set of Latin squares $L_1, ..., L_m$ is called *mutually orthogonal*, if for every $1 \leq i < j \leq m$, L_i and L_j are orthogonal. These are also referred to as *mutually orthogonal Latin squares* (MOLS).

4.1.3. Transversal designs (TDs)

A *transversal design* TD(k, n) of order (or group size) n and block size k is a triple $(\mathcal{P}, \mathcal{G}, \mathcal{B})$, where

(1) $\mathcal{P}$ is a set of kn points.

(2) $\mathcal{G}$ is a partition of $\mathcal{P}$ into k classes of size n, called *groups*.

(3) $\mathcal{B}$ is a collection of k-subsets of $\mathcal{P}$, called *blocks*.

(4) Every unordered pair of points from $\mathcal{P}$ is contained either in exactly one group or in exactly one block (cf. [79]).

It follows from (1)-(4) that any point of $\mathcal{P}$ occurs in exactly n blocks and that $|\mathcal{B}| = n^2$. Furthermore, axiom (4) implies that every block of $\mathcal{B}$ consists of exactly one point per group. Hence, by identifying the groups with k different colors, we obtain a k-coloring of the set system $(\mathcal{P}, \mathcal{B})$.

Theorem 9 *For $k \geq 3$, the existence of a set of $m := k - 2$ mutually orthogonal Latin squares (MOLS) of order n is equivalent to the existence of a TD(k, n) [79, 98, 99].*

Proof. We will outline the proof of this known result, since it is important for the understanding of the thesis. Let $L_1, \ldots, L_m$ be m MOLS with symbol sets $S_1, \ldots, S_m$, and with common row and column sets X and Y, respectively. We may assume that the sets $X, Y, S_1, \ldots, S_m$ are pairwise disjoint, which can easily be achieved by renaming the elements. Then we obtain a TD(k, n) with points $\mathcal{P} = \{X \cup Y \cup S_1 \cup \ldots \cup S_m\}$, groups $\mathcal{G} = \{X, Y, S_1, \ldots, S_m\}$ and blocks $\mathcal{B} = \{\{x, y, L_1[x, y], \ldots, L_m[x, y]\} : (x, y) \in X \times Y\}$. This process can be reversed to recover a set of $m = k - 2$ MOLS from a TD(k, n) for $k \geq 3$. ■

4.2. LDPC codes based on cyclic-structured TDs

In this chapter, our focus lies on an infinite family of structured TD LDPC codes that are highly amenable for extensive mathematical analysis and thus are ideally suited for analyzing and eliminating stopping sets and absorbing sets. First, we will have a closer look at the construction of these codes and their properties.

4.2.1. Construction

(1) We start by choosing a prime power order q which defines a finite field $\mathbb{F}_q$. This parameter solely determines the resulting code length by $N = q^2$.

(2) Next, we construct an m-set of mutually orthogonal Latin squares (MOLS) where m can be chosen between 1 and $q-2$. This parameter determines the constant column weight of the code's parity-check matrix by $k = m + 2$. The i-th Latin square of the set of MOLS is a $q \times q$ array filled with q symbols such that the cell indexed by row $x \in \mathbb{F}_q$ and column $y \in \mathbb{F}_q$ contains the symbol $\alpha_i x + \beta_i y \in \mathbb{F}_q$ for the scale factors $\alpha_i, \beta_i \in \mathbb{F}_q \setminus \{0\}$. In order to obtain MOLS, the scale factors $(\alpha_1, \beta_1), \ldots, (\alpha_m, \beta_m)$ of the m Latin squares have to be chosen in such a way that $\alpha_i \beta_j \neq \alpha_j \beta_i$ over $\mathbb{F}_q$ for $1 \leq i, j \leq m$ with $i \neq j$. As we can observe, each Latin square has a simple cyclic structure in that sense that each row is obtained from the previous one by adding the scale factor α_i component-wise over $\mathbb{F}_q$. The same holds for the rows by adding the scale factor β_i.

(3) In the next step, the set of m MOLS of order q is transformed into a transversal design of order q and block size $k = m + 2$. This relation has originally been described in [100, 99] and is outlined in Thm. 9.

(4) Finally, the incidence matrix of the transversal design is directly used as the parity-check matrix of an LDPC code, termed $\mathscr{L}_q^m$*–TD LDPC code*. We say that the order q of the underlying MOLS and the scale factors $\alpha_1, \ldots, \alpha_m$ are the *code parameters* which describe the code uniquely.

4.2.2. Properties

The basic parameters of an $\mathscr{L}_q^m$–TD LDPC code can be expressed in terms of q and m. More precisely, the code is $(m+2, q)$-regular, has length $N = q^2$, a number of $M = q(m+2)$ parity-check equations and code rate $R \geq \frac{q-m-2}{q}$. Experimental results show that the 2-rank of the code's parity-check matrix is precisely $q(m+2) - m - 1$ and thus is nearly full, even though this has not been rigorously proven. This leads to a code rate of exactly $R = \frac{q^2-q(m+2)+m+1}{q^2}$ which is slightly larger than the design rate $R_d = \frac{q^2-q(m+2)}{q^2} = \frac{q-m-2}{q}$. Furthermore, the factor graphs of the resulting codes are free of harmful 4-cycles and have a girth of 6.

We have shown in [P2, Thm. 4] that every set of MOLS with associated scale factors $\{(\alpha_i, \beta_i) : 1 \leq i \leq m\}$ can be reduced into an isomorphic form of MOLS with scale factors $\{(\alpha_i, 1) : 1 \leq i \leq m\}$ such that both sets of MOLS lead to the same code. Consequently, one scale factor per Latin square is theoretically sufficient for generating the entire set of possible $\mathscr{L}_q^m$–TD LDPC codes for a given order q. However, the second scale factor can be utilized to transfer the parity-check matrix of these codes into a form with quasi-cyclic structure which enables low-complexity encoding by feedback shift registers [21]. This process has been elaborated in our paper [P2, Section VI].

In the remainder of this chapter, we only consider $\mathscr{L}_q^m$–TD LDPC codes based on MOLS in reduced form, i.e., with associated scale factors $\{(\alpha_i, 1) : 1 \leq i \leq m\}$. This consideration is sufficient for the purpose of analyzing the stopping and absorbing set spectra of the resulting codes. It is crucial to notice that for a fixed set of parameters (q, m), there exists a large series of non-isomorphic $\mathscr{L}_q^m$–TD LDPC codes depending on the choice of the scale factors $\alpha_1, \ldots, \alpha_m$. These codes share the same basic parameters but may exhibit significantly different decoding performances. The following proposition enumerates the number of possible choices for the scale factors for fixed q and m which lead to codes with non-isomorphic parity-check matrices. The results are visualized in Fig. 4.1.

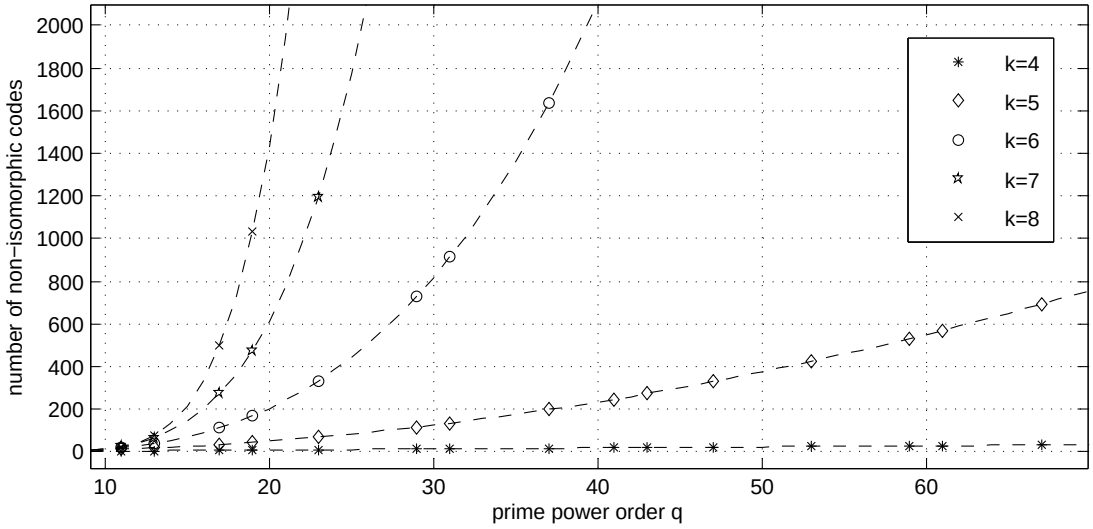


Abbildung 4.1. – Number of non-isomorphic $\mathscr{L}_q^m$–TD LDPC codes in terms of the prime power order q and column weight $k = m + 2$.

Proposition 10 *For every prime power order q and column weight $k = m + 2$ with $1 \leq m \leq q - 2$, there exist $\left\lceil \binom{q-1}{k-2}/(q-1) \right\rceil$ different $\mathscr{L}_q^m$–TD LDPC codes whose parity-check matrices are non-isomorphic.*

Proof. For column weight k, we need $k - 2$ MOLS. We assume w.l.o.g. that these MOLS are in reduced form [P3, Thm. 4] and thus we may associate the i-th Latin square with a single scale factor $\alpha_i \in \mathbb{F}_q^*$. Since all scale factors must be distinct, there are $\binom{q-1}{k-2}$ possible choices for $\alpha_1, \ldots, \alpha_{k-2}$. Finally, it follows from [P3, Thm. 5] that $q - 1$ choices produce the same code. The ceiling function is needed for the pathological case when $\lambda \odot \{\alpha_1, \ldots, \alpha_m\} = \{\alpha_1, \ldots, \alpha_m\}$ for a $\lambda \in \mathbb{F}_q \setminus \{0, 1\}$ and for at least one choice of scale factors, where $\odot$ is the componentwise multiplication. ■

Example. Let $q = 5$ and $m = 2$, then there are six possible choices for the scales factors $\{\alpha_1, \alpha_2\}$, namely $\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}$ and $\{3, 4\}$. By [P3, Thm. 4], we know that the MOLS based on the scale factors $\big\{\lambda \odot \{1, 2\} : \lambda \in \mathbb{F}_5^*\big\} = \big\{\{1, 2\}, \{1, 3\}, \{2, 4\}, \{3, 4\}\big\}$ leads to the same code where $\odot$ is the componentwise multiplication, as

well as MOLS based on the scale factors $\{\lambda \odot \{1,4\} : \lambda \in \mathbb{F}_5^*\} = \{\{1,4\},\{2,3\}\}$. Consequently, there are two different $\mathscr{L}_5^2$–TD LDPC codes with non-isomorphic parity-check matrices. These both codes are $(4,5)$-regular, have length $N = 25$, a number of $M = 20$ parity-check equations and code rate $R = 0.32$.

It is noted that the construction of MOLS is an old problem that has already been addressed from a purely mathematical standpoint. In particular, the method used to generate the cyclic-structured MOLS over finite fields can be found in [101]. This method has been adapted in [27] in order to construct TD LDPC codes of arbitrary column weights, but their further investigations are primarily focused on codes of column weight three that avoid codewords of weight four (known as *Pasch-configurations* from a design theoretic perspective, e.g., [102, 103]). In our papers [P2, P3], we have generalized this construction by using two scale factors for each Latin square. Although one scale factor per Latin square is theoretically sufficient for generating the entire set of possible $\mathscr{L}_q^m$–TD LDPC codes, the second scale factor is essential for transferring the parity-check matrix into quasi-cyclic structure [P2, Section VI] which allows low-complexity encoding.

4.3. Stopping set analysis

This section provides an extensive stopping set analysis of TD LDPC codes, in particular, for the infinite family of $\mathscr{L}_q^m$–TD LDPC codes described in the previous section. The focus lies primarily on codes of column weight 3 and 4, but the results are also potentially useful for higher column weights. Due to the simplified structure of the underlying transversal designs, these codes provide an excellent algebraic framework for the investigation of stopping sets. Recall that the family of $\mathscr{L}_q^m$–TD LDPC codes consists of subseries of $\lceil \binom{q-1}{k-2}/(q-1) \rceil$ different codes for each prime power order q and column weight k. Although the codes within the same series have identical basic parameters, they exhibit entirely different decoding performances over the BEC and thus reveal great potential for optimization. The main objective is therefore to identify those code instan-

ces from each subseries that have the most beneficial stopping set distributions. For this, we have derived powerful conditions that only depend on the scale factors of the underlying MOLS as well as on the choice of the prime power order q. By satisfying these conditions, we obtain high-rate TD LDPC codes with excellent decoding performances over the BEC under erasure (or peeling) decoding [53, 104], in particular, with very low error-floors.

As a first step, the results of our publication [P2] are shortly summarized as a basis for further investigations. The paper provides an exhaustive examination of stopping sets in LDPC codes from transversal designs. For this, we have transferred the concept of stopping sets to the level of Latin squares [P2, Section III], leading to the notion of full-correlating subrectangles [P2, Subsection III-B]. Basically, these subrectangles can be thought of as a simplified representation of stopping sets in terms of the underlying MOLS which can easily be analyzed. More specifically, each cell of a subrectangle obeys the same calculation rule given by the linear equation $\alpha_i x + y$ for the cell at position (x, y) of the i-th Latin square with scale factor α_i. Hence, the stopping sets can be described by systems of linear equations in a closed algebraic form. By solving these systems symbolically, i.e., in dependence of the scale factors and the order q, we can derive exact conditions for the existence of the corresponding stopping sets.

Based on a theoretical analysis, we have exactly determined the stopping distance of $\mathscr{L}_q^1$–TD LDPC code of column weight three [P2, Thm. 5] and lower bounded the stopping distance of $\mathscr{L}_q^2$–TD LDPC code of column weight four [P2, Thm. 6]. The results are outlined in the following proposition.

Proposition 11 *Let $H(m, q)$ be the parity-check matrix of an $\mathscr{L}_q^m$–TD LDPC codes of column weight $k = m + 2$ and let ω_q be the characteristic of the finite field $\mathbb{F}_q$ such that $q = \omega_q^t$ for any $t \geq 1$. Then, it holds that*

$$s_{min}\big(H(1,q)\big) = \begin{cases} 4, & \text{if } \omega_q = 2, \\ 6, & \text{if } \omega_q > 2, \end{cases} \tag{4.1}$$

$$s_{min}\big(H(2,q)\big) \geq \begin{cases} 6, & \text{if } \omega_q = 2 \text{ or } 3, \\ 8, & \text{if } \omega_q > 3. \end{cases} \tag{4.2}$$

Furthermore, by extensive computer simulations, we have revealed only two types of full-correlating subrectangles that may occur in any pair of MOLS from $\mathscr{L}_q^2$ for $\omega_q > 3$ [P2, Subsection V-A]. These subrectangles correspond to stopping sets of size eight in the resulting $\mathscr{L}_q^2$–TD LDPC codes and it is shown that they can be eliminated by satisfying the conditions $\alpha_1 + \alpha_2 \neq 0$, $2\alpha_1 - \alpha_2 \neq 0$ and $\alpha_1 - 2\alpha_2 \neq 0$ over $\mathbb{F}_q$. Furthermore, no stopping sets of size nine has been found. As a consequence, it ca be conjectured that the stopping distance can be raised from eight to ten for the case of $\omega_q > 3$.

In [P2, Subsection V-B], we describe a technique to construct full-correlating subrectangles by the composition of translations. By generalizing this method to higher m, we obtain full-correlating subrectangles in MOLS from $\mathscr{L}_q^m$ by the composition of full-correlating subrectangles in MOLS from $\mathscr{L}_q^{m-1}$. Consequently, we can construct unavoidable stopping sets for every $\mathscr{L}_q^m$–TD LDPC code, leading to the following proposition.

Proposition 12 *Let H be the parity-check matrix of an $\mathscr{L}_q^m$–TD LDPC code based on a set of m cyclic MOLS and let H' be the parity-check matrix of an $\mathscr{L}_q^{m-1}$–TD LDPC code based on an $(m-1)$-subset of these MOLS. Then, every stopping set of size ℓ in H' leads to a stopping set of size at most 2ℓ in H. In particular, it follows that*

$$s_{min}(H) \leq 2 s_{min}(H'). \tag{4.3}$$

Combined with equation (4.1), we obtain for the case of $\mathscr{L}_q^2$–TD LDPC codes that

$$s_{\min}\big(H(2,q)\big) \leq \begin{cases} 8, & \text{if } \omega_q = 2, \\ 12, & \text{if } \omega_q > 2. \end{cases} \tag{4.4}$$

In [P2, Subsection V-C], we have developed a technique for the analytical optimization of $\mathscr{L}_q^2$–TD LDPC codes by maximizing the size of the full-correlating subrectangles that arise from the composition of translations. As a result, we can significantly reduce the

number of stopping sets of size ten in $\mathscr{L}_q^2$–TD LDPC codes if the conditions $\alpha_1+\alpha_2 \neq 0$, $\alpha_1 - 2\alpha_2 \neq 0$, $2\alpha_1 - \alpha_2 \neq 0$ and $\alpha_1^2 - \alpha_1\alpha_2 + \alpha_2^2 \neq 0$ are satisfied over $\mathbb{F}_q$. Finally, we have identified further types of stopping sets of size 10 that may occur in $\mathscr{L}_q^2$–TD LDPC codes. These stopping sets can partially be eliminated by satisfying the conditions $\alpha_1^2+\alpha_1\alpha_2-\alpha_2^2 \neq 0$, $\alpha_2^2+\alpha_1\alpha_2-\alpha_1^2 \neq 0$ and $\alpha_1^2-3\alpha_1\alpha_2+\alpha_2^2 \neq 0$ over $\mathbb{F}_q$ [P2, Subsection V-D].

Based on the results of our publication [P2], the present chapter provides an enhanced and more detailed stopping set analysis of the infinite family of $\mathscr{L}_q^m$–TD LDPC codes (Section 4.2). The approach for this analysis is threefold. First, an exhaustive classification of the stopping set candidates for general TD LDPC codes is elaborated. Second, based on this classification, the specific structure of $\mathscr{L}_q^m$–TD LDPC codes is exploited in order to eliminate the most harmful stopping sets of these codes whenever this is possible and, if not, to provide an exact count of the stopping sets. In the third and last step, powerful design strategies are derived in order to obtain high-rate TD LDPC codes with excellent performances over the BEC under standard erasure decoding.

4.3.1. Stopping set candidates (SSCs)

From a coding theoretic perspective, stopping sets are defined as special subgraphs of a code's factor graph. However, from a design theoretic viewpoint, it is more natural to consider stopping sets as special set systems, allowing a more comfortable description of the combinatorial properties in terms of the underlying transversal design. Furthermore, stopping sets are, by definition, inextricably linked with a concrete code since they are subgraphs of one specific factor graph. This viewpoint is problematic when we investigate the existence of stopping sets since it does not admit their absence. Consequently, Def. 13 introduces the notion of stopping set candidates in the form of set systems which are independent of any specific code instance.

Definition 13 *A stopping set candidate (SSC) for the family of TD LDPC codes of column weight k is a set system consisting of points and blocks that fulfil the following combinatorial conditions:*

(A) All blocks contain exactly k points,

(B) any two blocks have at most one point in common,

(C) the points can be partitioned into k classes such that every block has exactly one point per class, i.e., the set system is k-colorable, and

(D) every point is contained in at least two blocks.

The first three conditions (A)-(C) are clearly inherited from the underlying TD and the fourth condition (D) arise from the definition of a stopping set. Note that the partition of points into k classes is equivalent to the existence of a k-coloring of the SSC. This equivalence can be clarified by considering that the points of each TD are, by definiton, partitioned into k groups such that each block has exactly one point per group. Hence, by coloring the points within a group with the same color and each group with a different color, we obtain a k-coloring of the SSC for which the points of every block are colored uniquely.

Intuitively, an SSC can be seen as a pattern for a stopping set that satisfies all necessary combinatorial conditions in order to represent a stopping set in any TD LDPC code. The presence or absence of such a stopping set in a specific TD LDPC code finally depends on the code's structure. Also notice that an SSC can equivalently be represented by an incidence matrix such that the points and blocks of the SSC correspond to the rows and columns of this matrix. This matrix representation is used for the visualization of the SSCs in the following classification.

4.3.2. Classification of SSCs

This section gives an extensive classification of stopping set candidates that potentially occur in TD LDPC codes. The SSCs are uniquely labeled by $S_k^{(\ell)}\{i\}$, where ℓ is the

size of the SSC (number of blocks), k is the constant number of points per block and where the postfix $\{i\}$ enumerates those SSCs that have the same ℓ and k. Appendix B thoroughly describes the classification process in order to identify the most harmful SSCs that potentially occur in a TD LDPC code.[2] Based on this process, the current section presents an exhaustive collection of SSCs of column weights three and four up to size seven and eleven, respectively. The arising lists can easily be extended to larger SSCs, but this is deliberately avoided in order to keep the lists small and also, since larger stopping sets are known to be harmless compared to smaller ones.

Fig. C.1 in Appendix C shows the complete list of SSCs of size $\ell \leq 7$ that may occur in a TD LDPC code of column weight three. This collection arises from the procedure

$$\mathcal{O} := \textbf{SSC\_Classification}(3, 7).$$

Note that the resulting list of SSCs is consistent with the list of full subrectangles presented in [P2, Fig. 2], but since several of these subrectangles are isomorphic, they represent the same stopping set.

Fig. C.2 gives a complete list of all possible SSCs of size $\ell \leq 7$ that may occur in a TD LDPC code of column weight four, i.e., all SSCs obtained by

$$\mathcal{O} := \textbf{SSC\_Classification}(4, 7).$$

It can be observed that there are no SSCs of size smaller than five and of size seven.

For block sizes of eight and larger, our procedure outputs an extremly large number of SSCs of which the most ones do not occur in any $\mathscr{L}_q^2$–TD LDPC code due to their specific structure. Hence, in order to obtain a small and handy list of SSCs that occur in at least one $\mathscr{L}_q^2$–TD LDPC code, a different approach has been taken. For this, standard Monte carlo simulations has been performed for many $\mathscr{L}_q^2$–TD LDPC codes with different values of q and the occurring stopping sets has been detected and categorized. This approach is

[2] This process has originally been developed in our manuscript [P2] in order to classificate absorbing set candidates but it can easily be adapted for the classification of SSCs.

non-deterministic and non-exhaustive such the arising lists of SSCs may be incomplete although it is highly probable that all relevant stopping sets has been detected.

Fig. C.3 lists all SSCs of size eight and nine that has been detected for at least one $\mathscr{L}_q^2$–TD LDPC code of column weight four and order $q \geq 5$. It should be noted that there has been found more SSCs of size eight and nine that specifically occur in $\mathscr{L}_q^2$–TD LDPC codes for $\omega_q = 2$ or 3. However, these codes contain smaller and more dominant stopping sets of size four in the case of $\omega_q = 2$ and stopping sets of size six in the case of $\omega_q = 3$ which are therefore the main contributors to the error-floor performance of these codes. Hence, the influence of stopping sets of size eight and nine are less important and can be excluded from consideration.

Finally, Fig. C.4 and Fig. C.5 visualize all SSCs of size ten and eleven, respectively, which has been detected for at least one $\mathscr{L}_q^2$–TD LDPC code of column weight four with $\omega_q \geq 13$. In order to keep the lists small, all SSCs are excluded that specifically occur in $\mathscr{L}_q^2$–TD LDPC codes with $\omega_q < 13$. As we will see later, these codes have dominant stopping sets smaller than ten such that the influence of the stopping sets of size ten or larger is small such that they can be neglected.

4.3.3. Elimination of SSCs

Based on the presented classification of stopping set candidates, this section investigates the existence of each SSC in an $\mathscr{L}_q^m$–TD LDPC code by exploiting the special structure of the code. Once it is known, under which conditions these SSCs exist and how they can be eliminated, we can derive powerful conditions for the proper choice of code parameters to reveal the codes with the most beneficial stopping set distributions.

More specifically, we calculate the exact elimination condition for every SSC. This condition determines for any choice of q and the scale factors $\alpha_1, \ldots, \alpha_m$ if the SSC is present or absent in the corresponding $\mathscr{L}_q^m$–TD LDPC code. For calculating the exact elimination condition of an SSC, we use the elimination process outlined in Appendix E.

Tabelle 4.1. – List of important conditions over $\mathbb{F}_q$

Label	Condition	Label	Condition
A_1	$\omega_q \neq 2$	B_6	$\alpha_1^2 - 3\alpha_1\alpha_2 + \alpha_2^2 \neq 0$
A_2	$\omega_q \neq 3$	B_7	$\alpha_1^2 - \alpha_1\alpha_2 + \alpha_2^2 \neq 0$
A_3	$\omega_q \neq 5$	B_8	$\alpha_1 + 3\alpha_2 \neq 0$
A_4	$\omega_q \neq 7$	B_9	$3\alpha_1 + \alpha_2 \neq 0$
A_5	$\omega_q \neq 13$	B_{10}	$3\alpha_1 - 2\alpha_2 \neq 0$
A_6	$\omega_q \neq 17$	B_{11}	$2\alpha_1 - 3\alpha_2 \neq 0$
B_1	$\alpha_1 + \alpha_2 \neq 0$	B_{12}	$\alpha_1 + 2\alpha_2 \neq 0$
B_2	$2\alpha_1 - \alpha_2 \neq 0$	B_{13}	$2\alpha_1 + \alpha_2 \neq 0$
B_3	$\alpha_1 - 2\alpha_2 \neq 0$	B_{14}	$\alpha_1 - 3\alpha_2 \neq 0$
B_4	$\alpha_1^2 + \alpha_1\alpha_2 - \alpha_2^2 \neq 0$	B_{15}	$3\alpha_1 - \alpha_2 \neq 0$
B_5	$\alpha_2^2 + \alpha_1\alpha_2 - \alpha_1^2 \neq 0$	B_{16}	$\alpha_1^2 + \alpha_1\alpha_2 + \alpha_2^2 \neq 0$

This process has originally been presented in our publication [P3] for eliminating absorbing sets but it can equivalently be used for the case of SSCs. For the smallest and most harmful SSCs, the exact number of their occurrences is calculated additionally. These cardinalities are extremly useful in order to estimate the decoding performances over the BEC under iterative decoding. For calculating the exact cardinalities, the process has been extended to, simply spoken, count the solutions of the linear equation systems that give rise to a stopping set. It is worth noting here that this algorithm could only be implemented efficiently by exploiting the simple structure of $\mathscr{L}_q^m$–TD LDPC codes.

Let $\mathcal{S}$ be any SSC of our classification in the previous section and let $\mathcal{C}$ be an $\mathscr{L}_q^m$–TD LDPC code. We say that a stopping set of $\mathcal{C}$ is of *type* $\mathcal{S}$ if its set system representation is isomorphic to $\mathcal{S}$. When it is written that a stopping set is of *type* $\mathcal{S}_\varphi$, then it is meant that there occurs a stopping set of type $\mathcal{S}$ in $\mathcal{C}$ such that the given k-coloring φ of $\mathcal{S}$ complies with the partitioning of the points into the groups of the underlying TD. The function $e(\mathcal{S}_\varphi, \mathcal{C})$ gives the exact condition for the elimination of all stopping sets of type $\mathcal{S}_\varphi$ such that there does not occur any stopping set of type $\mathcal{S}_\varphi$ in $\mathcal{C}$ if and only if $e(\mathcal{S}_\varphi, \mathcal{C})$ is satisfied. Furthermore, the term $|\mathcal{S}_\varphi, \mathcal{C}|$ counts the number of stopping sets of type $\mathcal{S}_\varphi$ in $\mathcal{C}$. Clearly, it holds that $|\mathcal{S}_\varphi, \mathcal{C}| = 0$ if and only if $e(\mathcal{S}_\varphi, \mathcal{C}) = 1$. Note that both

Tabelle 4.2. – Let $\mathcal{C}$ be any $\mathscr{L}_q^1$–TD LDPC code of column weight three. For each SSC of Fig. C.1, the table presents all possible 3-colorings φ, the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$ and the exact cardinalities $|\mathcal{S}_\varphi, \mathcal{C}|$. The referenced subterms A_i are listed in Table 4.1. $\overline{\text{S}}$ means the negation of S and $f(\text{S}) := 1$ if S is satisfied, else 0.

SSC $\mathcal{S}$	3-colorings φ	$e(\mathcal{S}_\varphi, \mathcal{C})$	$\lvert\mathcal{S}_\varphi, \mathcal{C}\rvert$
$S_3^{(4)}$	$\{1,6\},\{2,5\},\{3,4\}$	A_1	$f(\overline{A_1})\frac{1}{4}q^2(q-1)$
$S_3^{(6)}\{1\}$	$\{1,6,7\},\{2,5,8\},\{3,4,9\}$	A_1	$f(\overline{A_1})\frac{1}{2}q^2(q-1)(q-2)$
$S_3^{(6)}\{2\}$	$\{1,6,9\},\{2,5,8\},\{3,4,7\}$	no	$\frac{1}{6}q^2(q-1)(q-2)$
$S_3^{(6)}\{3\}$	$\{1,8\},\{2,5,6\},\{3,4,7\}$	A_2	$f(\overline{A_2})\frac{1}{2}q^2(q-1)$
$S_3^{(7)}\{1\}$	$\{1,8,9\},\{2,5,7\},\{3,4,6\}$	A_1	$f(\overline{A_1})\frac{1}{2}q^2(q-1)(q-2)$
$S_3^{(7)}\{2\}$	$\{1,8,9\},\{2,5,7\},\{3,4,6\}$	always	0
$S_3^{(7)}\{3\}$	$\{1,8,9\},\{2,5,7\},\{3,4,6\}$	$\overline{A_1}$	$f(A_1)\frac{1}{2}q^2(q-1)$

Tabelle 4.3. – Let $\mathcal{C}$ be any $\mathscr{L}_q^2$–TD LDPC code of column weight four. For each SSC of Fig. C.2, the table presents all possible 4-colorings φ, the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$ and the exact cardinalities $|\mathcal{S}_\varphi, \mathcal{C}|$. The referenced subterms A_i and B_i are listed in Table 4.1. $\overline{\text{S}}$ means the negation of S and $f(\text{S}) := 1$ if S is satisfied, else 0.

SSC $\mathcal{S}$	4-colorings φ	$e(\mathcal{S}_\varphi, \mathcal{C})$	$\lvert\mathcal{S}_\varphi, \mathcal{C}\rvert$
$S_4^{(6)}\{1\}$	$\{1,9,10\},\{2,6,12\},\{3,7,8\},\{4,5,11\}$	$A_1 \vee B_{16}$	$f(\overline{A_1} \wedge \overline{B_{16}})q^2(q-1)$
$S_4^{(6)}\{2\}$	$\{1,11\},\{2,6,10\},\{3,7,8\},\{4,5,9\}$	$A_2 \vee B_1$	$f(\overline{A_2} \wedge \overline{B_1})\frac{2}{3}q^2(q-1)$

functions are calculated in dependence of the code parameters q and $\alpha_1, \ldots, \alpha_m$ (scale factors) such that the results are valid for all $\mathscr{L}_q^m$–TD LDPC codes.

For each SSC of Fig. C.1, Table 4.2 gives their exact elimination conditions for $\mathscr{L}_q^1$–TD LDPC codes of column weight three with respect to their 3-colorings. Furthermore, their exact cardinalities are listed, meaning the number of their occurrences in an $\mathscr{L}_q^m$–TD LDPC code. For the SSCs of Fig. C.2, Table 4.3 gives all possible 4-colorings, their exact elimination conditions and their cardinalities. Finally, Table 4.4 presents all non-isomorphic 4-colorings of the SSCs classificated in Fig. C.3, Fig. C.4 and Fig. C.5, as well as the exact conditions for their elimination in $\mathscr{L}_q^2$–TD LDPC codes.

Tabelle 4.4. – Let $\mathcal{C}$ be any $\mathscr{L}_q^2$–TD LDPC code of column weight four. For each SSC of Fig. C.3, C.4 and C.5, the table presents all non-isomorphic 4-colorings φ and the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$. The referenced subterms A_i and B_i are listed in Table 4.1. $\overline{\text{S}}$ is the negation of S and $f(\text{S}) := 1$ if S is satisfied, else 0.

SSC $\mathcal{S}$	4-colorings φ	$e(\mathcal{S}_\varphi, \mathcal{C})$
$S_4^{(8)}\{1\}$	$\{1,10,11,16\},\{2,7,9,12\},\{3,6,8,13\},\{4,5,14,15\}$	$B_1 \wedge B_2 \wedge B_3$
	$\{1,8,13,16\},\{2,7,9,12\},\{3,6,10,11\},\{4,5,14,15\}$	A_1
$S_4^{(8)}\{2\}$	$\{1,8,12,14\},\{2,5,13\},\{3,6,9,11\},\{4,7,10\}$	$A_3 \vee (B_1 \wedge B_2 \wedge B_3)$
$S_4^{(9)}\{1\}$	$\{1,8,13,16\},\{2,5,9,11\},\{3,6,10,14\},\{4,7,12,15\}$	$A_3 \vee (B_1 \wedge B_2 \wedge B_3)$
$S_4^{(9)}\{2\}$	$\{1,8,12,16\},\{2,5,10,13\},\{3,6,9,14\},\{4,7,11,15\}$	$A_4 \vee (B_1 \wedge B_2 \wedge B_3)$
$S_4^{(9)}\{3\}$	$\{1,11,14,16\},\{2,5,8,12\},\{3,6,9,13\},\{4,7,10,15\}$	$A_4 \vee (B_{12} \wedge B_{13})$
$S_4^{(10)}\{1\}$	$\{1,8,12,17,19\},\{2,5,9,11,15\},\{3,6,10,13,18\},\{4,7,14,16\}$	$\overline{A_1} \vee \overline{A_2} \vee \overline{A_3} \vee (B_1 \wedge B_2 \wedge B_3)$
	5 other non-isomorphic 4-colorings	always
$S_4^{(10)}\{2\}$	$\{1,9,12,13,18\},\{2,6,11,16,19\},\{3,5,8,14,17\},\{4,7,10,15\}$	$A_5 \vee (B_1 \wedge B_2 \wedge B_3)$
$S_4^{(10)}\{3\}$	$\{1,8,14,17,19\},\{2,5,9,11\},\{3,6,12,16,18\},\{4,7,10,13,15\}$	$\overline{A_1} \vee \overline{A_2} \vee \overline{A_3} \vee (B_1 \wedge B_2 \wedge B_3)$
	5 other non-isomorphic 4-colorings	always
$S_4^{(10)}\{4\}$	$\{1,8,13,16,19\},\{2,5,9,15,17\},\{3,6,10,11\},\{4,7,12,14,18\}$	$\overline{A_2} \vee \overline{A_3} \vee \overline{A_4} \vee (B_1 \wedge B_2 \wedge B_3)$
	8 other non-isomorphic 4-colorings	always
$S_4^{(10)}\{5\}$	$\{1,9,12,15,20\},\{2,7,10,11,19\},\{3,6,8,13,18\},\{4,5,14,16,17\}$	$B_4 \wedge B_5 \wedge B_6$
	$\{1,10,12,17,20\},\{2,7,8,16,19\},\{3,6,9,11,13\},\{4,5,14,15,18\}$	B_7
	$\{1,10,12,18,19\},\{2,7,8,15,20\},\{3,6,9,11,13\},\{4,5,14,16,17\}$	$\overline{A_1} \vee \overline{B_1} \vee \overline{B_2} \vee \overline{B_3} \vee \overline{B_7}$
$S_4^{(10)}\{6\}$	$\{1,8,15,18\},\{2,5,9,17\},\{3,6,11,13,16\},\{4,7,10,12,14\}$	$A_5 \vee (B_8 \wedge B_9)$
$S_4^{(10)}\{7\}$	$\{1,9,13,18\},\{2,7,10,12,15\},\{3,6,11,16,17\},\{4,5,8,14,19\}$	$\overline{A_1} \vee \overline{B_1} \vee B_7$
$S_4^{(10)}\{8\}$	$\{1,8,13,17\},\{2,5,14,15\},\{3,6,9,11,16\},\{4,7,10,12\}$	$\overline{A_1} \vee \overline{A_2} \vee (B_1 \wedge B_2 \wedge B_3)$
	2 other non-isomorphic 4-colorings	always
$S_4^{(11)}\{1\}$	$\{1,8,15,18,19\},\{2,5,9,13,14\},\{3,6,12,16,17\},\{4,7,10,11,20\}$	$\overline{A_1} \vee \overline{A_2} \vee \overline{A_3} \vee A_5$ $\vee (B_1 \wedge B_2 \wedge B_3)$
	$\{1,9,10,13,14\},\{2,7,11,19,20\},\{3,6,12,16,17\},\{4,5,8,15,18\}$	always
$S_4^{(11)}\{2\}$	$\{1,9,12,13,20\},\{2,5,8,14,16\},\{3,6,10,15,17\},\{4,7,11,18,19\}$	$A_5 \vee (B_1 \wedge B_2 \wedge B_3)$
	$\{1,9,11,14,18\},\{2,7,8,13,19\},\{3,6,10,15,17\},\{4,5,12,16,20\}$	always
$S_4^{(11)}\{3\}$	$\{1,8,14,17,20\},\{2,5,10,12,15\},\{3,6,9,13,16\},\{4,7,11,18,19\}$	$\overline{A_1} \vee \overline{B_1} \vee \overline{B_2} \vee \overline{B_3} \vee \overline{B_7}$
	3 other non-isomorphic 4-colorings	always
$S_4^{(11)}\{4\}$	$\{1,11,15,18,20\},\{2,5,8,13,19\},\{3,6,9,14,16\},\{4,7,10,12,17\}$	$A_5 \vee (B_{10} \wedge B_{11} \wedge B_{12}$ $\wedge B_{13} \wedge B_{14} \wedge B_{15})$
$S_4^{(11)}\{5\}$	$\{1,9,12,15,18\},\{2,6,11,14,17\},\{3,5,8,13,19\},\{4,7,10,16\}$	$\overline{A_1} \vee B_7 \vee \overline{B_1}$
$S_4^{(11)}\{6\}$	$\{1,8,13,17,20\},\{2,5,11,14,16\},\{3,6,9,12,19\},\{4,7,10,15,18\}$	$A_5 \vee (B_{10} \wedge B_{11} \wedge B_{12} \wedge B_{13})$
$S_4^{(11)}\{7\}$	$\{1,11,15,16,20\},\{2,5,8,12,18\},\{3,6,9,13,17\},\{4,7,10,14,19\}$	$A_6 \vee (B_1 \wedge B_2 \wedge B_3)$

Example. Consider the SSC $S_3^{(4)}$ which has exactly one 3-coloring. We can eliminate all stopping sets of type $S_3^{(4)}$ in an $\mathscr{L}_q^1$–TD LDPC codes if the condition A_1 is satisfied. Conversely, if A_1 is not satisfied, then there exist $\frac{1}{4}q^2(q-1)$ stopping sets of type $S_3^{(4)}$ whose points are divided into groups by the underlying TD in the same way as the points of $S_3^{(4)}$ are partitioned into the color classes by the given 3-coloring.

4.4. Design strategies over the binary erasure channel

This section serves to investigate the implications of our stopping set analysis to the design of $\mathscr{L}_q^m$–TD LDPC codes and to derive new and powerful strategies in order to obtain codes with the most beneficial stopping set distributions. The arising conditions are expressed in terms of the order q and the scale factors $\alpha_1, \ldots, \alpha_m$ of the underlying set of MOLS from $\mathscr{L}_q^m$. Furthermore, let ω_q be the characteristic of $\mathbb{F}_q$, i.e., $q = \omega_q^t$ for any $t \geq 1$. Before the design strategies are presented in detail, two propositions are formulated which are very important for the elaboration of our strategies.

Proposition 14 *Let H the parity-check matrix of an $\mathscr{L}_p^m$–TD LDPC code based on MOLS with prime order p and let H' be the parity-check matrix of an $\mathscr{L}_{p^t}^m$–TD LDPC code based on MOLS with the same choice of scale factors, but with prime power order p^t for any $t > 0$. Then, every stopping set in H leads directly to a stopping set in H' of the same size but not vice versa. In particular, it holds that $s_{min}(H') \leq s_{min}(H)$.*

Proof. Consider H and H' as the incidence matrix of a transversal design $\mathcal{D}$ and $\mathcal{D}'$, respectively. It can be shown that $\mathcal{D}$ has a large number of subdesigns of type $\mathcal{D}'$ (cf. [105]) and thus every stopping set of H directly results in a stopping set of H' of the same size. The converse does not hold in general. ■

Proposition 15 *Let H be the parity-check matrix of an $\mathscr{L}_q^m$–TD LDPC code and let H' be the parity-check matrix of an $\mathscr{L}_q^{m-t}$–TD LDPC code for $1 \leq t \leq m-1$ such that*

H' arises from H by deleting the rows corresponding to any t groups of the underlying $\mathscr{L}_q^m$–TD. Then, every stopping set in H directly leads to a stopping set in H' of the same size but not vice versa. In particular, it holds that $s_{min}(H') \leq s_{min}(H)$.

Proof. Consider the matrix representation of any stopping set in H. From the definition of stopping sets we know that every row of this matrix must have a weight of at least 2. This property does not get lost by deleting any rows of H such that the matrix with potentially fewer rows remains a stopping set in H'. Conversely, the property can be destroyed by adding rows of weight one such that a stopping set in H' does not generally result in a stopping set of H. ■

Strategy 1 *The order q of an $\mathscr{L}_q^m$–TD LDPC code should be a large prime number.*

Proof. This strategy is motivated by the following arguments:

* The $\mathscr{L}_{p^t}^m$–TD LDPC codes of length $N = p^{2t}$ based on MOLS with prime power order p^t for $t > 1$ should be avoided since their stopping distances are upper bounded by the stopping distances of the corresponding $\mathscr{L}_p^m$–TD LDPC codes of length $N = p^2$ based on MOLS of order p with the same scale factors (Proposition 14). Hence, the stopping distance does not increase for larger code lengths.
* There exist stopping sets that specifically occur for a certain characteristic ω_q and thus are termed ω_q-*specific* stopping sets. It can be observed that the size of the smallest ω_q-specific stopping set tends to be smaller if ω_q is small. For instance, if $\omega_q = 2$, the $\mathscr{L}_2^1$–TD LDPC codes of column weight three exclusively contain the smallest 2-specific stopping sets of type $S_3^{(4)}$ and for $\omega_q = 3$, the smallest 3-specific stopping sets are of type $S_3^{(6)}\{3\}$. For the case of $\mathscr{L}_2^2$–TD LDPC codes of column weight four, the smallest 2-specific stopping sets are of type $S_4^{(6)}\{1\}$, the smallest 3-specific stopping sets of type $S_4^{(6)}\{2\}$, the smallest 5-specific stopping sets of type $S_4^{(8)}\{2\}$, the smallest 7-specific stopping sets of type $S_4^{(9)}\{2\}$ and $S_4^{(9)}\{3\}$, the smallest 13-specific stopping sets of type $S_4^{(10)}\{2\}$ and $S_4^{(10)}\{6\}$, and finally, the smallest 17-specific stopping sets of type $S_4^{(11)}\{7\}$. As we can see, the size of the

smallest ω_q-specific stopping sets steadily grows from 6 to 11 for $\omega_q = 2$ to 17. Furthermore, it can be expected that there are even larger ω_q-specific stopping sets for $\omega_q > 17$. Hence, in order to eliminate these types of stopping sets, it is reasonable to choose q as a large prime number which avoids small characteristics.

■

For covering the whole spectrum of $\mathscr{L}_q^m$–TD LDPC codes, the following strategies are nevertheless presented in terms of a general prime power order q.

Strategy 2 (Design of $\mathscr{L}_q^1$–TD LDPC codes of column weight three)
Based on our investigations, we obtain the best $\mathscr{L}_q^1$–TD LDPC codes over the BEC via standard erasure decoding by choosing the prime power order q such that $\omega_q > 3$ where ω_q is the characteristic of $\mathbb{F}_q$. The arising codes have a stopping distance of six.

Proof. This strategy is based on the following arguments:

* The stopping distance can be raised from four to six by choosing $\omega_q \neq 2$ such that all stopping sets of type $S_3^{(4)}$ are eliminated. This is the maximal achievable stopping distance, since there always occur size-6 stopping sets of type $S_3^{(6)}\{2\}$.
* The number of the smallest stopping sets of size six can be minimized by satisfying $\omega_q > 3$ such that stopping sets of type $S_3^{(6)}\{1\}$, $S_3^{(6)}\{3\}$ and $S_3^{(6)}\{4\}$ are eliminated.
* The eliminated SSCs are most likely contained in larger stopping sets which can trivially be avoided in addition.

■

Strategy 3 (Design of $\mathscr{L}_q^2$–TD LDPC codes of column weight four)
Based on our investigations, we obtain the best $\mathscr{L}_q^2$–TD LDPC codes over the BEC via standard erasure decoding by choosing the prime power order q and the scale factors α_1 and α_2 in such a way that the conditions A_1 to A_6 and B_1 to B_7 are satisfied over $\mathbb{F}_q$. The arising codes have a stopping distance of ten.

Proof. This strategy relies on the following arguments:

* The smallest and most harmful stopping sets of types $S_4^{(6)}\{1\}$ and $S_4^{(6)}\{2\}$ can be eliminated by satisfying the conditions A_1 and A_2.
* All size-8 stopping sets of type $S_4^{(8)}\{1\}$ and $S_4^{(8)}\{2\}$ can be avoided by A_1 and B_1 to B_3. Furthermore, all size-9 stopping sets of type $S_4^{(9)}\{1\}$ to $S_4^{(9)}\{3\}$ can be eliminated by satisfying A_3 and A_4.
* Hence, we can raise the stopping distance from six to ten by fulfilling the conditions A_1 to A_4 and B_1 to B_3.
* The stopping distance for our properly designed $\mathscr{L}_q^2$–TD LDPC codes is exactly ten since there occur stopping sets of type $S_4^{(10)}\{5\}$. This is the largest possible stopping distance for any $\mathscr{L}_q^2$–TD LDPC code of column weight four.
* The number of dominating stopping sets of size ten can be reduced significantly by satisfying B_1 to B_7 and A_5. More precisely, the SSCs $S_4^{(10)}\{1\}$ to $S_4^{(10)}\{4\}$ can be eliminated by B_1 to B_3 and the SSCs $S_4^{(10)}\{6\}$ to $S_4^{(10)}\{8\}$ can be avoided by A_5, B_1 to B_3 and B_7. Furthermore, the number of stopping sets of type $S_4^{(10)}\{5\}$ can be significantly reduced by satisfying B_4 to B_7.
* The number of stopping sets of size eleven can be reduced substantially by satisfying the conditions A_5, A_6 and B_7. More precisely, the proposed strategy eliminates all stopping sets of type $S_4^{(11)}\{1\}$ to $S_4^{(11)}\{7\}$ except stopping sets of type $S_4^{(11)}\{3\}$.

■

Strategy 4 (Design of $\mathscr{L}_q^m$–TD LDPC codes of column weight five and higher) *For $m > 2$, we obtain excellent $\mathscr{L}_q^m$–TD LDPC codes of column weight five and higher by choosing the prime power order q and the scale factors $\alpha_1, \ldots, \alpha_m$ in such a way that the conditions A_1 to A_5 and B_1 to B_7 are satisfied over $\mathbb{F}_q$.*

Proof. This strategy is motivated by the following considerations:

* There are two possibilites in order to improve the stopping set distribution of $\mathscr{L}_q^m$–TD LDPC codes of higher column weights. First, we can carry out an exact

stopping set analysis of these codes and derive a design strategy based on this analysis just as we have done it for the codes of column weight $k \leq 4$. Unfortunately, this approach leads to a very high computational complexity for growing column weights such that an exhaustive analysis is extremly expensive. Second, we can exploit the topological relations between the stopping set distributions of $\mathscr{L}_q^m$–TD LDPC codes and those of smaller column weights such that the design strategy for $\mathscr{L}_q^m$–TD LDPC codes with higher column weights can be reduced to the strategies elaborated for codes of smaller column weights. This second approach is the basis of the current strategy and will subsequently be described in more detail.

* Consider an $\mathscr{L}_q^m$–TD LDPC code with parity-check matrix H and an $\mathscr{L}_q^{m-t}$–TD LDPC code of lower column weight with parity-check matrix H' that arises from H by deleting all rows that correspond to an arbitrary selection of t groups of the underlying TD. As we know from Prop. 15, every stopping set in H leads to a stopping set in H' of the same size but with reduced column weight. Hence, the elimination of stopping sets in $\mathscr{L}_q^{m-t}$–TD LDPC codes greatly supports the elimination of stopping sets in $\mathscr{L}_q^m$–TD LDPC codes of higher column weight.

■

Based on the presented design strategies, Table 4.5 shows the parameters of possible $\mathscr{L}_q^2$–TD LDPC codes of column weight four for every prime order q up to 100. As we can see, our strategies admit a wide range of possible codes for every prime order q.

Tabelle 4.5. – For the family of $\mathscr{L}_q^2$–TD LDPC codes of column weight $k = 4$, the table lists the best possible choices for the scale factors α_1, α_2 for prime orders q up to 100 in order to obtain excellent codes with maximum stopping distance $s_{min}(H) = 10$ according to the design strategies elaborated in Section 4.4 over the BEC. Note that there are no convenient choices for $q < 19$.

q	Define $\alpha_1 = 1$. Then, the best choices for α_2 are	N	R
19	$\{2, \ldots, 18\} \setminus \{2, 4, 5, 6, 8, 10, 12, 14, 15, 16, 18\}$	361	0.80
23	$\{2, \ldots, 22\} \setminus \{2, 12, 22\}$	529	0.83
29	$\{2, \ldots, 28\} \setminus \{2, 5, 6, 7, 15, 23, 24, 25, 28\}$	841	0.87
31	$\{2, \ldots, 30\} \setminus \{2, 6, 12, 13, 14, 16, 18, 19, 20, 26, 30\}$	961	0.87
37	$\{2, \ldots, 36\} \setminus \{2, 11, 19, 27, 36\}$	1369	0.89
41	$\{2, \ldots, 40\} \setminus \{2, 6, 7, 8, 21, 34, 35, 36, 40\}$	1681	0.90
43	$\{2, \ldots, 42\} \setminus \{2, 7, 22, 37, 42\}$	1849	0.91
47	$\{2, \ldots, 46\} \setminus \{2, 24, 46\}$	2209	0.92
53	$\{2, \ldots, 52\} \setminus \{2, 27, 52\}$	2809	0.93
59	$\{2, \ldots, 58\} \setminus \{2, 25, 26, 27, 30, 33, 34, 35, 58\}$	3481	0.93
61	$\{2, \ldots, 60\} \setminus \{2, 14, 17, 18, 19, 31, 43, 44, 45, 48, 60\}$	3721	0.94
67	$\{2, \ldots, 66\} \setminus \{2, 30, 34, 38, 66\}$	4489	0.94
71	$\{2, \ldots, 70\} \setminus \{2, 8, 9, 10, 36, 62, 63, 64, 70\}$	5041	0.94
73	$\{2, \ldots, 72\} \setminus \{2, 9, 37, 65, 72\}$	5329	0.95
79	$\{2, \ldots, 78\} \setminus \{2, 24, 29, 30, 31, 40, 49, 50, 51, 56, 78\}$	6241	0.95
83	$\{2, \ldots, 82\} \setminus \{2, 42, 82\}$	6889	0.95
89	$\{2, \ldots, 88\} \setminus \{2, 9, 10, 11, 45, 79, 80, 81, 88\}$	7921	0.96
97	$\{2, \ldots, 96\} \setminus \{2, 36, 49, 62, 96\}$	9409	0.96

4.5. Absorbing set analysis

In this chapter, we thoroughly investigate and design $\mathscr{L}_q^m$–TD LDPC codes for their optimal usage over the AWGN channel under iterative decoding with the standard sum-product algorithm. The presented results are based on our manuscript [P3], but the findings of this paper are extended significantly. The current chapter provides a comprehensive absorbing set analysis by a threefold approach. First, an exhaustive classification of possible absorbing sets is presented which may occur in the factor graph of any TD LDPC code. Second, based on this classification, the specific structure of $\mathscr{L}_q^m$–TD LDPC codes is exploited for the elimination of the most harmful absorbing sets. Third, powerful conditions are derived in order to identify those codes with the most beneficial absorbing set spectra. As a result, we obtain an infinite family of high-rate regular LDPC codes with excellent performances over the AWGN channel, in particular, with very low-error floors.

4.5.1. Absorbing set candidates (ASCs)

By definition, absorbing sets are special subgraphs of a code's factor graph. From a design theoretic viewpoint, it is more natural to consider absorbing sets as set systems which allow a more comfortable description of the combinatorial properties in terms of the underlying transversal design. Furthermore, absorbing sets are inextricably linked with a concrete code since they are subgraphs of one specific factor graph. This viewpoint is problematic when we investigate the existence of absorbing sets since it does not admit their absence. Consequently, Def. 16 introduces the notion of absorbing set candidates which are described in the form of set systems independent of a specific code instance.[3]

[3] Although the following definition of absorbing set candidates is tailored to the family of TD LDPC codes, they must not necessarily occur in any code of this family.

Definition 16 *An absorbing set candidate (ASC) for the family of TD LDPC codes with column weight k is a set system of points and blocks such that*

(A) each block contains exactly k points,

(B) any two blocks share at most one point,

(C) the points can be partitioned into classes such that every block has exactly one point per class, i.e., the set system is k-colorable, and

(D) for every block, the majority of points from this block is contained in an even number of blocks.

The first three conditions (A)-(C) are imposed by the properties of a transversal design and the fourth condition arise from the definition of an absorbing set. In particular, note, that property (C) directly follows from the property of a TD that its point set is partitioned in k groups such that every block of the TD has exactly one point per group. As for absorbing sets, the *size* of an ASC shall be the number of blocks and the *syndrome* shall be defined as the number of points with odd degree.

Intuitively, absorbing set candidates can be seen as patterns for absorbing sets that satisfies all necessary combinatorial conditions in order to represent an absorbing set in TD LDPC codes, but which must not necessarily occur in a specific code instance of this code family. An ASC can equivalently be represented by its binary incidence matrix such that the points and blocks of the ASC correspond to the rows and columns of the matrix. In the next section, this matrix representation is primarily used for the visualization of the ASCs.

4.5.2. Classification of ASCs

This section presents an extensive classification of small absorbing set candidates that potentially occur in TD LDPC codes. The ASCs of the classification are uniquely labeled by $(a, b)_k\{i\}$, where a is the size of the ASC (number of blocks), b is the syndrome (number of points with odd degree), k is the constant block size and where the postfix

$\{i\}$ enumerates those ASCs that have the same size, syndrome and block size. The applied classification process is based on the procedure developed in [P3, Section III] and is outlined in Appendix B. Basically, by starting with an empty set system, the current set system is successively extended in all possible ways in compliance with certain combinatorial constraints that must be satisfied in order to represent an ASC. Note that the specific structure of TD LDPC codes significantly reduces the number of ways in which a set system can be extended since only k-colorable set systems are admitted. Hence, we obtain a narrow collection of ASCs precisely tailored for the family of TD LDPC codes.

Table D.1 and Table D.2 show a complete classification of ASCs for TD LDPC codes of column weight three and four, respectively, up to size six. Note that these tables have originally been presented in [P3, Subsection III-B and C]. For the case of TD LDPC codes with column weight four, the classification is expanded by a complete list of all 34 ASCs of size eight and syndrome of at most two in Table D.3. Note that this table also visualizes the inclusions between the ASCs. Furthermore, an exhaustive classification of ASCs up to size six for TD LDPC codes of column weight five is given in Table D.4. For keeping the list small, the size-6 ASCs with syndromes larger than six has been excluded which are supposed to be harmless since they have relatively large syndromes. It is worth noting here that the classification is suitable for the entire family of TD LDPC codes and not only for a specific code within this family, whereas, in contrast, many search algorithms for finding small absorbing sets (e.g. [106, 107]) result in lists of absorbing sets that occur in the factor graph of one specific code.

4.5.3. Elimination of ASCs

Based on the presented classification of ASCs in the previous section, we are now concerned with their existence in $\mathscr{L}_q^m$–TD LDPC codes. As will be demonstrated, some of the associated absorbing sets can be eliminated or partially avoided by a proper choice of the code parameters, more precisely, by the right choice of the scale factors α_i and the

prime power order q of the underlying finite field. Finally, we derive powerful conditions for the most convenient choices under the objective of avoiding the potentially most harmful absorbing sets.

First, we will define some basic notations. Let $\mathcal{S}$ be any ASC and let $\mathcal{C}$ be an $\mathscr{L}_q^m$–TD LDPC code. We say that an absorbing set of $\mathcal{C}$ is of *type* $\mathcal{S}$ if the set system representation of the absorbing set is isomorphic to $\mathcal{S}$. Furthermore, let φ be any k-coloring of $\mathcal{S}$, where k is the block size of $\mathcal{S}$. Then, an absorbing set is of *type* $\mathcal{S}_\varphi$, if, in addition, the given k-coloring complies with the partition of the points into groups by the underlying transversal design. Furthermore, we have shown in [P3, Thm. 2] that an absorbing set of type $\mathcal{S}_\varphi$ is fully if all points of $\mathcal{S}$ with an odd degree are contained in exactly one color class of φ. The converse does also hold for the cases of $k = 3$ and 4.

We now investigate the existence of an ASC $\mathcal{S}$ in dependence of the code parameters of $\mathcal{C}$. For this, we have developed a procedure in [P3, Section V] in order to calculate the exact conditions for which absorbing sets of type $\mathcal{S}$ exist or can be eliminated. This procedure exploits the specific structure of $\mathscr{L}_q^m$–TD LDPC codes and is thoroughly described and outlined in Appendix E. Basically, every ASC can be described by a linear equation system in a closed algebraic form. This system is then solved symbolically by a modified Gaussian elimination algorithm in dependence of the code parameters.

By applying the described procedure on every ASC of the classification presented in Section 4.5.2, we obtain the exact conditions of how the respective absorbing sets can be eliminated or partially avoided in dependence of the code parameters of $\mathcal{C}$. The results are presented in several tables by giving the following information for each ASC $\mathcal{S}$.

- First, all possible k-colorings of $\mathcal{S}$ are listed.
- For each coloring φ of $\mathcal{S}$, the term $e(\mathcal{S}_\varphi, \mathcal{C})$ gives the exact elimination condition, i.e., there is no absorbing set of type $\mathcal{S}_\varphi$ in $\mathcal{C}$ if and only if $e(\mathcal{S}_\varphi, \mathcal{C})$ is satisfied.
- For each coloring of $\mathcal{S}$, the term $|\mathcal{S}_\varphi, \mathcal{C}|$ gives the number of occurrences of absorbing sets of type $\mathcal{S}_\varphi$ in $\mathcal{C}$. Clearly, it holds that $|\mathcal{S}_\varphi, \mathcal{C}| = 0$ if and only if $e(\mathcal{S}_\varphi, \mathcal{C})$ is satisfied.

Tabelle 4.6. – List of important conditions over $\mathbb{F}_q$

Label	Condition	Label	Condition
C_1	$\alpha_1 + \alpha_2 \neq 0$	C_{19}	$\alpha_1^2 - 3\alpha_1\alpha_2 + 3\alpha_2^2 \neq 0$
C_2	$2\alpha_1 - \alpha_2 \neq 0$	C_{20}	$3\alpha_1 - 4\alpha_2 \neq 0$
C_3	$\alpha_1 - 2\alpha_2 \neq 0$	C_{21}	$4\alpha_1 - 3\alpha_2 \neq 0$
C_4	$\omega_q \neq 2$	C_{22}	$\alpha_1 - 4\alpha_2 \neq 0$
C_5	$\alpha_1^2 + \alpha_1\alpha_2 - \alpha_2^2 \neq 0$	C_{23}	$4\alpha_1 - \alpha_2 \neq 0$
C_6	$\alpha_2^2 + \alpha_1\alpha_2 - \alpha_1^2 \neq 0$	C_{24}	$\alpha_1 + 3\alpha_2 \neq 0$
C_7	$\alpha_1^2 - 3\alpha_1\alpha_2 + \alpha_2^2 \neq 0$	C_{25}	$3\alpha_1 + \alpha_2 \neq 0$
C_8	$\alpha_1^2 - \alpha_1\alpha_2 + \alpha_2^2 \neq 0$	C_{26}	$\alpha_1^2 + \alpha_2^2 \neq 0$
C_9	$\omega_q \neq 3$	C_{27}	$2\alpha_1^2 - 2\alpha_1\alpha_2 + \alpha_2^2 \neq 0$
C_{10}	$3\alpha_1 - 2\alpha_2 \neq 0$	C_{28}	$\alpha_1^2 - 2\alpha_1\alpha_2 + 2\alpha_2^2 \neq 0$
C_{11}	$2\alpha_1 - 3\alpha_2 \neq 0$	C_{29}	$\alpha_1^3 + \alpha_1^2\alpha_2 + \alpha_2^3 \neq 0$
C_{12}	$\alpha_1 + 2\alpha_2 \neq 0$	C_{30}	$\alpha_1^3 + \alpha_1\alpha_2^2 + \alpha_2^3 \neq 0$
C_{13}	$2\alpha_1 + \alpha_2 \neq 0$	C_{31}	$\alpha_1^2 - 2\alpha_2^2 \neq 0$
C_{14}	$\alpha_1 - 3\alpha_2 \neq 0$	C_{32}	$2\alpha_1^2 - \alpha_2^2 \neq 0$
C_{15}	$3\alpha_1 - \alpha_2 \neq 0$	C_{33}	$\alpha_1^2 - 4\alpha_1\alpha_2 + 2\alpha_2^2 \neq 0$
C_{16}	$\alpha_1^2 + \alpha_1\alpha_2 + \alpha_2^2 \neq 0$	C_{34}	$2\alpha_1^2 - 4\alpha_1\alpha_2 + \alpha_2^2 \neq 0$
C_{17}	$\omega_q \neq 5$	C_{35}	$\alpha_1^2 + 2\alpha_1\alpha_2 - \alpha_2^2 \neq 0$
C_{18}	$3\alpha_1^2 - 3\alpha_1\alpha_2 + \alpha_2^2 \neq 0$	C_{36}	$\alpha_2^2 + 2\alpha_1\alpha_2 - \alpha_1^2 \neq 0$

Tabelle 4.7. – Let $\mathcal{C}$ be any $\mathscr{L}_q^1$–TD LDPC code of column weight three. For each ASC of Fig. D.1, the table presents all possible 3-colorings φ, the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$ and the cardinalities $|\mathcal{S}_\varphi, \mathcal{C}|$. An asterisk $(*)$ indicates that an absorbing set of type $\mathcal{S}_\varphi$ is full. The subterms C_i are listed in Table 4.6. $\overline{\mathrm{S}}$ is the negation of S and $f(\mathrm{S}) := 1$ if S is satisfied, else 0.

ASC $\mathcal{S}$	3-coloring φ	$e(\mathcal{S}_\varphi, \mathcal{C})$	$\lvert\mathcal{S}_\varphi, \mathcal{C}\rvert$
$(3,3)_3$	$\{1,6\},\{2,5\},\{3,4\}$	no	$q^2(q-1)$
$(4,0)_3$	$\{1,6\},\{2,5\},\{3,4\}$ $(*)$	C_4	$f(\overline{C_4})\frac{1}{4}q^2(q-1)$
$(4,2)_3$	$\{1,6,7\},\{2,5\},\{3,4\}$ $(*)$	$\overline{C_4}$	$f(C_4)\frac{3}{2}q^2(q-1)$
$(4,4)_3$	$\{1,6\},\{2,5,8\},\{3,4,7\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)$
	$\{1,6\},\{2,4\},\{3,5,7,8\}$ $(*)$	no	$\leq \frac{3}{4}q^2(q-1)(q-2)$
$(5,3)_3\{1\}$	$\{1,6,7\},\{2,5,8\},\{3,4,9\}$	no	$\leq 3q^2(q-1)(q-2)$
$(5,3)_3\{2\}$	$\{1,6,9\},\{2,5,8\},\{3,4,7\}$	no	$q^2(q-1)(q-2)$
$(5,5)_3$	$\{1,7,9,10\},\{2,5,8\},\{3,4,6\}$	no	$\leq 3q^2(q-1)(q-2)(q-3)$
$(6,0)_3\{1\}$	$\{1,6,7\},\{2,5,8\},\{3,4,9\}$ $(*)$	C_4	$f(\overline{C_4})\frac{1}{2}q^2(q-1)(q-2)$
$(6,0)_3\{2\}$	$\{1,6,9\},\{2,5,8\},\{3,4,7\}$ $(*)$	no	$\frac{1}{6}q^2(q-1)(q-2)$
$(6,2)_3\{1\}$	$\{1,8\},\{2,5,6\},\{3,4,7\}$ $(*)$	C_9	$f(\overline{C_9})\frac{1}{2}q^2(q-1)$
$(6,2)_3\{2\}$	$\{1,8,9\},\{2,5,6\},\{3,4,7\}$ $(*)$	$\overline{C_4} \vee \overline{C_9}$	$\leq f(C_4 \wedge C_9, 3q^2(q-1))$
$(6,2)_3\{3\}$	$\{1,6,7\},\{2,5,9,10\},\{3,4,8\}$ $(*)$	$\overline{C_4}$	$\leq f(C_4)\frac{5}{2}q^2(q-1)(q-2)$
$(6,2)_3\{4\}$	$\{1,6,7\},\{2,5,8\},\{3,4,9,10\}$ $(*)$	$\overline{C_4}$	$\leq f(C_4)3q^2(q-1)(q-2)$
$(6,2)_3\{5\}$	$\{1,6,7,10\},\{2,5,8\},\{3,4,9\}$ $(*)$	$\overline{C_4}$	$\leq f(C_4)\frac{3}{2}q^2(q-1)(q-2)$
$(6,2)_3\{6\}$	$\{1,6,9,10\},\{2,5,8\},\{3,4,7\}$ $(*)$	always	0
$(6,4)_3\{1\}$	$\{1,10\},\{2,5,7,8\},\{3,4,6,9\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)$
$(6,4)_3\{2\}$	$\{1,8,9,10\},\{2,5,6\},\{3,4,7\}$ $(*)$	no	$\leq q^2(q-1)(q-2)$
$(6,4)_3\{3\}$	$\{1,6,8,10,11\},\{2,5,7\},\{3,4,9\}$ $(*)$	no	$\leq 3q^2(q-1)(q-2)(q-3)$
	$\{1,6,8,9\},\{2,5,7\},\{3,4,10,11\}$	no	$\leq 5q^2(q-1)(q-2)(q-3)$
$(6,4)_3\{4\}$	$\{1,6,8,10\},\{2,5,7,11\},\{3,4,9\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)(q-3)$
$(6,4)_3\{5\}$	$\{1,6,8\},\{2,5,9,10\},\{3,4,7,11\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)(q-3)$
	$\{1,7,9,10,11\},\{2,4,8\},\{3,5,6\}$ $(*)$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)(q-3)$
	$\{1,7,8,10\},\{2,4,9,11\},\{3,5,6\}$	no	$\leq \frac{5}{2}q^2(q-1)(q-2)(q-3)$
	$\{1,6,8\},\{2,4,9,11\},\{3,5,7,10\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)(q-3)$
$(6,4)_3\{6\}$	$\{1,6,8\},\{2,5,9,10\},\{3,4,7,11\}$	no	$\leq 3q^2(q-1)(q-2)(q-3)$
	$\{1,6,9,11\},\{2,4,8\},\{3,5,7,10\}$	no	$\leq \frac{5}{2}q^2(q-1)(q-2)(q-3)$
$(6,6)_3\{1\}$	$\{1,10,11\},\{2,5,7,8\},\{3,4,6,9\}$	no	$\leq \frac{3}{2}q^2(q-1)(q-2)(q-3)$
$(6,6)_3\{2\}$	$\{1,7,9,10\},\{2,5,8,11\},\{3,4,6,12\}$	no	$\leq \frac{1}{2}q^2(q-1)^2(q-2)(q-3)$
	$\{1,7,8,11\},\{2,5,9,10\},\{3,4,6,12\}$	no	$\leq \frac{3}{2}q^2(q-1)^2(q-2)(q-3)$
	$\{1,6,8\},\{2,5,9,11,12\},\{3,4,7,10\}$	no	$\leq \frac{5}{2}q^2(q-1)^2(q-2)(q-3)$
	$\{1,6,8\},\{2,4,10\},\{3,5,7,9,11,12\}$ $(*)$	no	$\leq \frac{1}{2}q^2(q-1)^2(q-2)(q-3)$

Tabelle 4.8. – Let $\mathcal{C}$ be any $\mathscr{L}_q^2$–TD LDPC code of column weight four. For each ASC of Fig. D.2, the table presents all non-isomorphic 4-colorings φ, the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$ and the cardinalities $|\mathcal{S}_\varphi, \mathcal{C}|$. An asterisk $(*)$ indicates that an absorbing set of type $\mathcal{S}_\varphi$ is full. The subterms C_i are listed in Table 4.6. $\overline{\mathrm{S}}$ is the negation of S and $f(\mathrm{S}) := 1$ if S is satisfied, else 0.

ASC $\mathcal{S}$	4-coloring φ	$e(\mathcal{S}_\varphi, \mathcal{C})$	$\lvert\mathcal{S}_\varphi, \mathcal{C}\rvert$
$(4,4)_4$	$\{1,8\},\{2,6\},\{3,7,9\},\{4,5,10\}$	$C_1 \wedge C_2 \wedge C_3$	$q^2(q-1)\sum_{\ell=1}^{3} f(\overline{C_\ell})$
	$\{1,8\},\{2,6\},\{3,5\},\{4,7,9,10\}$ $(*)$	C_4	$q^2(q-1)f(\overline{C_4})$
$(5,4)_4$	$\{1,9,10\},\{2,6,12\},\{3,7,8\},\{4,5,11\}$	$C_5 \wedge C_6 \wedge C_7$	$2q^2(q-1)\sum_{\ell=5}^{7} f(\overline{C_\ell})$
$(6,0)_4$	$\{1,9,10\},\{2,6,12\},\{3,7,8\},\{4,5,11\}$ $(*)$	$C_4 \vee C_{16}$	$q^2(q-1)f(\overline{C_4} \wedge \overline{C_{16}})$
$(6,2)_4\{1\}$	$\{1,11\},\{2,6,10\},\{3,7,8\},\{4,5,9\}$ $(*)$	$C_1 \vee C_9$	$\frac{2}{3}q^2(q-1)f(\overline{C_1} \wedge \overline{C_9})$
$(6,2)_4\{2\}$	$\{1,11,12\},\{2,6,10\},\{3,7,8\},\{4,5,9\}$ $(*)$	always	0
$(6,2)_4\{3\}$	$\{1,8,12,13\},\{2,6,11\},\{3,7,9\},\{4,5,10\}$ $(*)$	$(C_1 \wedge C_2 \wedge C_3) \vee \overline{C_9}$	$2q^2(q-1)\sum_{\ell=1}^{3} f(\overline{C_\ell} \wedge C_9)$
	$\{1,8,11\},\{2,6,12,13\},\{3,7,9\},\{4,5,10\}$ $(*)$	always	0
$(6,2)_4\{4\}$	$\{1,9,10\},\{2,6,12,13\},\{3,7,8\},\{4,5,11\}$ $(*)$	always	0
$(6,4)_4\{1\}$	$\{1,11,12,13\},\{2,6,10\},\{3,7,8\},\{4,5,9\}$ $(*)$	$C_8 \vee \overline{C_1}$	$\frac{4}{3}q^2(q-1)f(\overline{C_8} \wedge C_1)$
$(6,4)_4\{2\}$	$\{1,9,11,13,14\},\{2,6,12\},\{3,7,8\},\{4,5,10\}$ $(*)$	$C_4 \vee \overline{C_{16}}$	$12q^2(q-1)f(\overline{C_4} \wedge C_{16})$
	$\{1,9,11,12\},\{2,6,13,14\},\{3,7,8\},\{4,5,10\}$	always	0
	$\{1,9,10,13\},\{2,6,12\},\{3,7,8\},\{4,5,11,14\}$	$(C_{10} \wedge C_{11} \wedge C_{12} \wedge C_{13} \wedge C_{14} \wedge C_{15}) \vee \overline{C_{17}}$	$\sum_{\ell=10}^{15} f(\overline{C_\ell} \wedge C_{17}, 4q^2(q-1))$
	$\{1,8,10\},\{2,6,12\},\{3,7,9,13\},\{4,5,11,14\}$	$C_9 \vee \overline{C_1}$	$12q^2(q-1)f(\overline{C_9} \wedge C_1)$
	$\{1,8,10\},\{2,7,11,13\},\{3,5,12\},\{4,6,9,14\}$	$C_8 \vee \overline{C_1} \vee \overline{C_4}$	$12q^2(q-1)f(\overline{C_8} \wedge C_1 \wedge C_4)$
	$\{1,8,10\},\{2,6,13,14\},\{3,5,12\},\{4,7,9,11\}$	always	0
	$\{1,8,10\},\{2,6,12\},\{3,5,13,14\},\{4,7,9,11\}$	$C_{10} \wedge C_{11} \wedge C_{12} \wedge C_{13} \wedge C_{14} \wedge C_{15}$	$2q^2(q-1)\sum_{\ell=10}^{15} f(\overline{C_\ell})$
$(6,4)_4\{3\}$	$\{1,9,10\},\{2,6,12\},\{3,7,8,14\},\{4,5,11,13\}$	$(C_{16} \wedge C_{18} \wedge C_{19}) \vee \overline{C_4}$	$2q^2(q-1)\sum_{\ell=16,18,19} f(\overline{C_\ell} \wedge C_4)$
	$\{1,9,10\},\{2,6,13,14\},\{3,5,12\},\{4,7,8,11\}$	$C_{20} \wedge C_{21} \wedge C_{22} \wedge C_{23} \wedge C_{24} \wedge C_{25}$	$q^2(q-1)\sum_{\ell=20}^{25} f(\overline{C_\ell})$
$(6,4)_4\{4\}$	$\{1,9,11,13,14\},\{2,6,12\},\{3,7,8\},\{4,5,10\}$ $(*)$	$\overline{C_1} \vee \overline{C_2} \vee \overline{C_3} \vee \overline{C_8}$	$4q^2(q-1)f(\overline{C_1} \wedge \overline{C_2} \wedge \overline{C_3} \wedge \overline{C_8})$
	$\{1,9,11,12\},\{2,6,13,14\},\{3,7,8\},\{4,5,10\}$	always	0
$(6,6)_4\{1\}$	$\{1,9,12,13\},\{2,7,11,14\},\{3,6,8\},\{4,5,10,15\}$	$C_1 \wedge C_2 \wedge C_3$	n/a
	$\{1,9,11,14\},\{2,7,12,13\},\{3,6,8\},\{4,5,10,15\}$	C_8	n/a
	$\{1,9,11,14\},\{2,7,12,13\},\{3,6,8\},\{4,5,10,15\}$	no	n/a
	$\{1,8,10\},\{2,7,11,14\},\{3,6,9,15\},\{4,5,12,13\}$	always	n/a
	$\{1,8,10\},\{2,7,12,13\},\{3,6,9,15\},\{4,5,11,14\}$	$(\overline{C_1} \vee \overline{C_4} \vee \overline{C_{14}}) \wedge (\overline{C_2} \vee \overline{C_4} \vee \overline{C_{11}} \vee \overline{C_{13}}) \wedge (\overline{C_3} \vee \overline{C_4} \vee \overline{C_{10}} \vee \overline{C_{12}})$	n/a
	$\{1,8,12,14\},\{2,6,11\},\{3,7,9,13\},\{4,5,10,15\}$	$C_1 \wedge (\overline{C_2} \vee \overline{C_3} \vee \overline{C_8} \vee \overline{C_9}) \vee (\overline{C_1} \wedge C_9)$	n/a
	$\{1,8,10\},\{2,6,11\},\{3,7,9,14,15\},\{4,5,12,13\}$	$C_1 \wedge C_2 \wedge C_3$	n/a
	$\{1,8,10\},\{2,6,11\},\{3,7,9,13\},\{4,5,12,14,15\}$	$(\overline{C_1} \vee \overline{C_4} \vee \overline{C_{14}} \vee \overline{C_{26}}) \wedge (\overline{C_2} \vee \overline{C_4} \vee \overline{C_{11}} \vee \overline{C_{13}} \vee \overline{C_{27}}) \wedge (\overline{C_3} \vee \overline{C_4} \vee \overline{C_{10}} \vee \overline{C_{12}} \vee \overline{C_{28}})$	n/a
	$\{1,8,10\},\{2,6,11\},\{3,5,13\},\{4,7,9,12,14,15\}$ $(*)$	C_4	n/a
$(6,6)_4\{2\}$	$\{1,10,12,14\},\{2,7,11,15\},\{3,6,8\},\{4,5,9,13\}$	C_4	n/a
	$\{1,10,11,15\},\{2,7,12,14\},\{3,6,8\},\{4,5,9,13\}$	$C_1 \wedge C_2 \wedge C_3$	n/a
	$\{1,9,11\},\{2,7,12,14\},\{3,6,8\},\{4,5,10,13,15\}$	always	n/a
	$\{1,8,12\},\{2,6,11\},\{3,5,9\},\{4,7,10,13,14,15\}$ $(*)$	no	n/a

Tabelle 4.9. – Let $\mathcal{C}$ be any $\mathscr{L}_q^2$–TD LDPC code of column weight four. For each ASC of Fig. D.3, the table presents all non-isomorphic 4-colorings φ and the exact elimination conditions $e(\mathcal{S}_\varphi, \mathcal{C})$. An asterisk $(*)$ indicates that an absorbing set of type $\mathcal{S}_\varphi$ is full. The subterms C_i are listed in Table 4.6. $\overline{\mathrm{S}}$ is the negation of S and $f(\mathrm{S}) := 1$ if S is satisfied, else 0.

ASC $\mathcal{S}$	4-coloring φ	$e(\mathcal{S}_\varphi, \mathcal{C})$
$(8,0)_4\{1\}$	$\{1,8,12,15\},\{2,6,13,14\},\{3,5,11,16\},\{4,7,9,10\}$ $(*)$	$C_4 \vee \overline{C_{16}}$
	$\{1,8,11,16\},\{2,6,12,15\},\{3,5,13,14\},\{4,7,9,10\}$ $(*)$	C_4
	3 other non-isomorphic 4-colorings	always
$(8,0)_4\{2\}$	$\{1,9,10,13\},\{2,6,12,15\},\{3,7,8,16\},\{4,5,11,14\}$ $(*)$	$(C_{29} \wedge C_{30}) \vee C_4$
$(8,0)_4\{3\}$	2 non-isomorphic 4-colorings	always
$(8,0)_4\{4\}$	$\{1,9,11,12\},\{2,6,13,14\},\{3,7,8,16\},\{4,5,10,15\}$ $(*)$	$(C_{29} \wedge C_{30}) \vee C_4$
	2 other non-isomorphic 4-colorings	always
$(8,0)_4\{5\}$	3 non-isomorphic 4-colorings	always
$(8,0)_4\{6\}$	$\{1,10,12,14\},\{2,7,11,15\},\{3,6,8,16\},\{4,5,9,13\}$ $(*)$	always
	$\{1,10,11,15\},\{2,7,12,14\},\{3,6,8,16\},\{4,5,9,13\}$ $(*)$	$C_1 \wedge C_2 \wedge C_3$
$(8,2)_4\{1\}$	$\{1,8,12,13\},\{2,6,11,14\},\{3,7,9,16,17\},\{4,5,10,15\}$ $(*)$	always
$(8,2)_4\{2\}$	4 non-isomorphic 4-colorings	always
$(8,2)_4\{3\}$	8 non-isomorphic 4-colorings	always
$(8,2)_4\{4\}$	5 non-isomorphic 4-colorings	always
$(8,2)_4\{5\}$	2 non-isomorphic 4-colorings	always
$(8,2)_4\{6\}$	$\{1,9,10,13,17\},\{2,6,12,14\},\{3,7,8,16\},\{4,5,11,15\}$ $(*)$	always
$(8,2)_4\{7\}$	$\{1,9,10,13\},\{2,6,12,15\},\{3,7,8,16,17\},\{4,5,11,14\}$ $(*)$	always
$(8,2)_4\{8\}$	2 non-isomorphic 4-colorings	always
$(8,2)_4\{9\}$	2 non-isomorphic 4-colorings	always
$(8,2)_4\{10\}$	$\{1,9,10,14\},\{2,6,12,16,17\},\{3,7,8,15\},\{4,5,11,13\}$ $(*)$	always
$(8,2)_4\{11\}$	$\{1,9,11,12\},\{2,6,13,14\},\{3,7,8,16,17\},\{4,5,10,15\}$ $(*)$	$(C_{29} \wedge C_{30}) \vee C_4$
$(8,2)_4\{12\}$	2 non-isomorphic 4-colorings	always
$(8,2)_4\{13\}$	6 non-isomorphic 4-colorings	always
$(8,2)_4\{14\}$	7 non-isomorphic 4-colorings	always
$(8,2)_4\{15\}$	$\{1,9,10,13\},\{2,6,12,16,17\},\{3,7,8,15\},\{4,5,11,14\}$ $(*)$	$((C_{10} \wedge C_{11}) \vee \overline{C_1}) \wedge ((C_{12} \wedge C_{14}) \vee \overline{C_2})$ $\wedge ((C_{13} \wedge C_{15}) \vee \overline{C_3})$
	8 other non-isomorphic 4-colorings	always
$(8,2)_4\{16\}$	$\{1,8,10,16,17\},\{2,7,11,12\},\{3,5,13,14\},\{4,6,9,15\}$ $(*)$	$(C_{27} \vee \overline{C_1} \vee \overline{C_3}) \wedge (C_{28} \vee \overline{C_1} \vee \overline{C_2})$ $\wedge (C_{26} \vee \overline{C_2} \vee \overline{C_3})$
	$\{1,8,10,16,17\},\{2,6,13,14\},\{3,5,12,15\},\{4,7,9,11\}$ $(*)$	$C_{31} \wedge C_{32} \wedge C_{33} \wedge C_{34}$ $\wedge C_{34} \wedge C_{35} \wedge C_{36}$
	3 other non-isomorphic 4-colorings	always
$(8,2)_4\{17\}$	3 non-isomorphic 4-colorings	always
$(8,2)_4\{18\}$	6 non-isomorphic 4-colorings	always
$(8,2)_4\{19\}$	6 non-isomorphic 4-colorings	always
$(8,2)_4\{20\}$	3 non-isomorphic 4-colorings	always
$(8,2)_4\{21\}$	6 non-isomorphic 4-colorings	always
$(8,2)_4\{22\}$	5 non-isomorphic 4-colorings	always
$(8,2)_4\{23\}$	4 non-isomorphic 4-colorings	always
$(8,2)_4\{24\}$	$\{1,8,11,12,17\},\{2,6,14,15\},\{3,5,13,16\},\{4,7,9,10\}$ $(*)$	$\overline{C_4} \vee \overline{C_9} \vee ((\overline{C_3} \vee \overline{C_{10}} \vee \overline{C_{12}})$ $\wedge (\overline{C_2} \vee \overline{C_9} \vee \overline{C_{13}}) \wedge (\overline{C_1} \vee \overline{C_{14}}))$
	5 other non-isomorphic 4-colorings	always
$(8,2)_4\{25\}$	3 non-isomorphic 4-colorings	always
$(8,2)_4\{26\}$	3 non-isomorphic 4-colorings	always
$(8,2)_4\{27\}$	3 non-isomorphic 4-colorings	always
$(8,2)_4\{28\}$	2 non-isomorphic 4-colorings	always

∗ For each coloring φ, the tables visualizes by an asterisk if the absorbing sets of type $\mathcal{S}_\varphi$ are fully in the case of their existence.

Table 4.7 gives the exact elimination conditions and cardinalities for the ASCs of Fig. D.1 which potentially occur in $\mathscr{L}_q^1$–TD LDPC codes of column weight three. Table 4.8 presents the exact elimination conditions and cardinalities for the ASCs of Fig. D.2 for $\mathscr{L}_q^2$–TD LDPC codes of column weight four. Finally, Table 4.9 gives the elimination conditions for the size-8 ASCs of Fig. D.3 for $\mathscr{L}_q^2$–TD LDPC codes of column weight four. The results for the ASCs of column weight five in Fig. D.4 are not presented in this section, but the implications for the strategy of designing $\mathscr{L}_q^3$–TD LDPC of column weight five are extensively discussed in Strategy 7. It is worth noting here that the elimination conditions and cardinalities of the ASCs are presented in terms of the prime power order q and the scale factors α_i and thus are valid for the entire family of $\mathscr{L}_q^m$–TD LDPC codes. Hence, the results can be used for the design of TD LDPC codes with an infinite range of code lengths and rates.

4.6. Design strategies over the AWGN channel

Based on the results of the absorbing set analysis, this section derives powerful strategies for the design of excellent $\mathscr{L}_q^m$–LDPC codes over the AWGN channel via standard SPA decoding. The main objective is to find the codes with the most beneficial absorbing set spectra from the family of $\mathscr{L}_q^m$–TD LDPC codes. However, the strategies should be as unrestricted as possible to maintain the parametric flexibility of the code family. Note that the presented results are valid for the entire range of possible $\mathscr{L}_q^m$–TD LDPC codes, whereas the results obtained by computer-based search algorithms are typically valid only for one specific parity-check matrix.

Strategy 5 (Strategy for $\mathscr{L}_q^1$-TD LDPC codes of column weight three)
Based on our results, we obtain the best $\mathscr{L}_q^1$–TD LDPC codes over the AWGN channel via standard SPA decoding by choosing the prime power order q of $\mathbb{F}_q$ in such a way that $\omega_q > 3$, where ω_q is the characteristic of $\mathbb{F}_q$.

Proof. The proof relies on the following arguments:

* The absorbing sets of type $(4,0)_3$ and $(6,0)_3\{1\}$ are eliminated. These absorbing sets are supposed to be the most harmful ones since they are very small, have syndrome 0 and are fully. In particular, the absorbing sets of type $(4,0)_3$ are the smallest possible fully absorbing sets and thus are extremly detrimental for the decoding process.
* The smallest absorbing sets of type $(3,3)_3$ are unavoidable, but they are relatively harmless since they are not full and have a relatively large syndrome of three.
* Unfortunately, the absorbing sets of type $(4,0)_3$ and $(4,2)_3$ can not be avoided simultaneously since their elimination depend on opposing conditions, but the $(4,0)_3$ absorbing sets are more harmful and thus have a higher priority.
* The absorbing sets of size 5 can not be avoided, but they are supposed to be harmless since they are not full and have relatively large syndromes.
* The $(6,0)_3\{1\}$ fully absorbing sets can be avoided by $\omega_q \neq 2$, whereas the $(6,0)_3\{2\}$ fully absorbing sets can generally not be avoided. Hence, the absorbing sets of type $(6,0)_3\{2\}$ are supposed to be the most harmful ones which mainly dominate the decoding performance in the error-floor region.
* Finally, the $(6,2)_3\{1\}$ absorbing sets can be avoided by $\omega_q \neq 3$ which reduces the number of absorbing sets of size six, whereas the $(6,2)_3\{2\}$, $(6,2)_3\{3\}$, $(6,2)_3\{4\}$ and $(6,2)_3\{5\}$ ones can not be avoided by $\omega_q > 3$.
* The absorbing sets of types $(6,4)\{i\}$ and $(6,6)\{i\}$ can not be eliminated, but they are supposed to be harmless due to their large syndromes.

■

Strategy 6 (Strategy for $\mathscr{L}_q^2$–TD LDPC codes of column weight four)
Based on our results, we obtain the best $\mathscr{L}_q^2$–TD LDPC codes over the AWGN channel via standard SPA decoding by choosing the prime power order q of $\mathbb{F}_q$ and the scale factors α_1 and α_2 in such a way that the conditions C_1 to C_{16} and C_{18} to C_{36} of Table 4.6 are satisfied.

Proof. The proof relies on the following arguments:

* The smallest possible absorbing sets have type $(4,4)_4$ and can be eliminated if and only if the conditions C_1 to C_4 of Table 4.6 are satisfied. The $(6,2)_4\{1\}$ and $(6,2)_4\{3\}$ absorbing sets can simultaneously be avoided since they contain a $(4,4)_4$ absorbing set.
* The smallest possible fully absorbing sets have size $(4,4)_4$ and can be eliminated if and only if C_4 is satisfied. Note that the absorbing sets of type $(4,4)_4$ are only full if the second 4-coloring listed in Table 4.8 is induced by the underlying TD.
* The smallest absorbing sets of syndrome 0 are of type $(6,0)_4$ and can be avoided if and only if $C_4 \vee C_{16}$ is satisfied. These absorbing sets are full and thus are extremly harmful for the decoding process. Note that $(6,0)_4$ absorbing sets correspond to codewords of minimum weight six and also define stopping sets of the same size. Hence, by avoiding these entities, we can also raise the minimum and stopping distance of the code.
* The absorbing sets of type $(5,4)_4$ can be avoided if and only if the conditions C_5 to C_7 are satisfied. Although these absorbing sets are supposed to be relatively harmless due to their large syndromes, they are contained in harmful $(6,0)_4$, $(6,2)_4\{4\}$, $(8,2)_4\{9\}$ and $(8,2)_4\{10\}$ absorbing sets (and most likely in larger ones) which can therefore be simultaneously be eliminated.
* The fully $(6,2)_4\{3\}$ absorbing sets can be avoided by satisfying the conditions C_1 to C_3 and the fully $(6,2)_4\{2\}$ and $(6,2)_4\{4\}$ absorbing sets do never occur due to the specific structure of the considered codes. Hence, all absorbing sets of size $a \leq 6$ and syndrome $b \leq 2$ can simultaneously be avoided by the proposed design strategy.
* The fully $(6,4)_4\{1\}$ absorbing sets can be avoided by satisfying the condition C_8.
* The $(6,4)_4\{2\}$ absorbing sets are eliminated by satisfying the conditions C_4 and C_8 to C_{15}, in particular, the subset of fully $(6,4)_4\{2\}$ absorbing sets are avoided by C_4. These absorbing sets are contained in harmful fully $(8,2)_4\{15\}$ ones which are therefore avoided concurrently. Also, they are contained in $(8,2)_4\{13\}$ and

$(8,2)_4\{14\}$ absorbing sets, but which can never occur.

* The $(6,4)_4\{3\}$ absorbing sets can be avoided by the conditions C_{16} and C_{18} to C_{25}. They are contained in $(8,2)_4\{6\}$ and $(8,2)_4\{25\}$ absorbing sets which are trivially eliminated.
* The fully $(6,4)_4\{4\}$ absorbing sets can not simultaneously be avoided by our design strategy since their elimination condition is opposed to the conditions of more harmful absorbing sets. Hence, the $(6,4)_4\{4\}$ absorbing sets are the smallest ones that definitely occur while pursuing our strategy. However, they have relatively large syndromes and thus are supposed to be rather harmless. Also, they are contained in absorbing sets of type $(8,2)_4\{5\}$ but which are trivially avoided.
* The absorbing sets of type $(6,6)_4\{1\}$ can partially be eliminated by C_1 to C_4 and $C8$. In particular, the fully $(6,6)_4\{1\}$ absorbing sets can be avoided if and only if the condition C_4 is satisfied. The absorbing sets of type $(6,6)_4\{2\}$ can partially be eliminated if the conditions C_1 to C_4 are satisfied, but the subset of fully $(6,6)_4\{2\}$ absorbing sets can not be avoided. Although the absorbing sets of types $(6,6)_4\{1\}$ and $(6,6)_4\{2\}$ are harmless due to their large syndromes, they are contained in $(8,0)_4\{4\}$, $(8,0)_4\{5\}$, $(8,0)_4\{6\}$, $(8,2)_4\{18\}$ and $(8,2)_4\{20\}$ to $(8,2)_4\{28\}$ absorbing sets and most likely in larger ones which can therefore partially be eliminated.
* The fully absorbing sets of types $(8,0)_4\{i\}$ with $i \in \{1,\ldots,6\}$ can all be eliminated by satisfying C_1 to C_4, C_{29} and C_{30}.
* The fully absorbing sets of types $(8,2)_4\{i\}$ with $i \in \{1,\ldots,28\}\setminus\{24\}$ are all simultaneously eliminated by the conditions C_{10} to C_{15} and C_{26} to C_{36}. The $(8,2)_4\{24\}$ absorbing sets are the only occurring ones in the codes obtained by our design strategy.

■

Strategy 7 (Strategy for $\mathscr{L}_q^3$–TD LDPC codes of column weight five)

We obtain the best $\mathscr{L}_q^3$–TD LDPC codes over the AWGN channel via standard SPA decoding by choosing the prime power order q of $\mathbb{F}_q$ and the scale factors α_1, α_2 and α_3 in such a way that the conditions D_1 to D_{26} of Table 4.10 are satisfied.

Tabelle 4.10. – List of conditions over $\mathbb{F}_q$ depending on the scale factors α_1, α_2 and α_3. A condition is satisfied, if it is satisfied for all possible permutations $(\alpha_x, \alpha_y, \alpha_z)$ of the given scale factors.

Label	Condition	Label	Condition
D_1	$\alpha_x - \alpha_y + \alpha_z \neq 0$	D_{15}	$\alpha_x\alpha_y^2 - \alpha_y^2\alpha_z + \alpha_x\alpha_z^2 - \alpha_x\alpha_y\alpha_z \neq 0$
D_2	$\alpha_x\alpha_y - \alpha_x\alpha_z + \alpha_y\alpha_z \neq 0$	D_{16}	$\alpha_x\alpha_y^2 - \alpha_y^2\alpha_z + \alpha_x^2\alpha_z - \alpha_x\alpha_y\alpha_z \neq 0$
D_3	$\alpha_x\alpha_y - 2\alpha_y\alpha_z + \alpha_z^2 \neq 0$	D_{17}	$\alpha_x\alpha_y^2 + \alpha_x^2\alpha_z + \alpha_y\alpha_z^2 - 3\alpha_x\alpha_y\alpha_z \neq 0$
D_4	$\alpha_x^2 - \alpha_y\alpha_z \neq 0$	D_{18}	$\alpha_x^2\alpha_y + \alpha_x^2\alpha_z + \alpha_y^2\alpha_z - 3\alpha_x\alpha_y\alpha_z \neq 0$
D_5	$\alpha_x + \alpha_y \neq 0$	D_{19}	$\alpha_x\alpha_z - 2\alpha_x\alpha_y + \alpha_y\alpha_z + \alpha_x^2 - \alpha_z^2 \neq 0$
D_6	$2\alpha_x - \alpha_y \neq 0$	D_{20}	$\alpha_x^2 + \alpha_y\alpha_z - 3\alpha_x\alpha_y + \alpha_y^2 \neq 0$
D_7	$2\alpha_x - \alpha_y - \alpha_z \neq 0$	D_{21}	$2\alpha_x^2\alpha_y - \alpha_x\alpha_y^2 - \alpha_x^2\alpha_z + \alpha_y^2\alpha_z - \alpha_x\alpha_y\alpha_z \neq 0$
D_8	$\alpha_x\alpha_y + \alpha_x\alpha_z - 2\alpha_y\alpha_z \neq 0$	D_{22}	$2\alpha_x^2\alpha_y - \alpha_x\alpha_y^2 - \alpha_x^2\alpha_z + \alpha_y\alpha_z^2 - \alpha_x\alpha_y\alpha_z \neq 0$
D_9	$\alpha_x\alpha_y - \alpha_x\alpha_z - \alpha_y^2 \neq 0$	D_{23}	$\alpha_x^2 + \alpha_x\alpha_y - \alpha_y^2 \neq 0$
D_{10}	$\alpha_x\alpha_y + \alpha_x\alpha_z - \alpha_y^2 \neq 0$	D_{24}	$\alpha_x^2 - 3\alpha_x\alpha_y + \alpha_y^2 \neq 0$
D_{11}	$\alpha_x\alpha_y - \alpha_y\alpha_z - \alpha_z^2 \neq 0$	D_{25}	$\alpha_x\alpha_z - 3\alpha_x\alpha_y + \alpha_y\alpha_z$
D_{12}	$\alpha_x^2 + \alpha_x\alpha_y - \alpha_x\alpha_z - \alpha_y^2 \neq 0$		$+\alpha_x^2 + \alpha_y^2 - \alpha_z^2 \neq 0$
D_{13}	$\alpha_x^2 - \alpha_x\alpha_y - \alpha_x\alpha_z + \alpha_z^2 \neq 0$	D_{26}	$\alpha_x^2\alpha_y^2 - \alpha_x^2\alpha_z^2 - \alpha_y^2\alpha_z^2 + 3\alpha_x\alpha_y\alpha_z^2$
D_{14}	$\alpha_x\alpha_y - 2\alpha_y\alpha_z - \alpha_x\alpha_z + \alpha_z^2 \neq 0$		$-\alpha_x\alpha_y^2\alpha_z - \alpha_x^2\alpha_y\alpha_z \neq 0$

Proof. The proof relies on the following arguments:

* The smallest possible absorbing sets are of type $(4, 8)_5$ and can be eliminated if and only if the conditions D_1 to D_8 of Table 4.10 are satisfied. Although these absorbing sets are probably not harmful themselves due to their large syndrome, their elimination is highly beneficial since they are contained in $(5, 5)_5$, $(5, 7)_5$, $(6, 0)_5$, $(6, 2)_5$, $(6, 4)_5\{1\}$, $(6, 4)_5\{2\}$ and $(6, 6)_5\{1\}$ to $(6, 6)_5\{4\}$ absorbing sets (and most likely in many larger ones) which can therefore be eliminated simultaneously. In particular, the $(6, 0)_5$, $(6, 2)_5$ absorbing sets are supposed to be the most harmful ones which are avoided by the proposed design strategy.
* The absorbing sets of type $(5, 9)_5$ can be avoided if the conditions D_1 to D_{26} are satisfied. These absorbing sets are relatively harmless due to their very large syndromes, but since they are contained in absorbing sets of type $(6, 4)_5\{2\}$ and $(6, 6)_5\{5\}$ (and most likely in even larger and more harmful ones), it is expected that their elimination improves the decoding performance in the error-floor region significantly.

* There are five non-isomorphic types of $(6,6)_5$ absorbing sets. Four of these types contain $(4,8)_5$ absorbing sets and one type contains a $(5,9)_5$ ASC and thus can be avoided simultaneously. Furthermore, there are eight non-isomorphic types of $(6,8)_5$ absorbing sets. Six of these types contain $(4,8)_5$ absorbing sets and two types contain $(5,9)_5$ absorbing sets. Hence, all possible absorbing sets of size $a \leq 6$ and syndrome $b \leq 8$ can be avoided by eliminating the $(4,8)_5$ and $(5,9)_5$ ones. The absorbing sets of size 6 and syndrome $b \geq 4$ are supposed to be relatively harmless, but they are most likely contained in larger and more harmful absorbing sets with smaller syndromes such that their elimination is highly advantageous.
* Finally, there are six non-isomorphic types of $(6,10)_5$ absorbing sets and two non-isomorphic types of $(6,12)_5$ absorbing sets. These types have not been evaluated due to the large computational complexity, but since their syndromes are very large, they are supposed to be harmless for the decoding process.

■

Based on the presented design strategies over the AWGN channel, Table 4.11 and Table 4.12 show the parameters of possible $\mathscr{L}_q^2$–TD LDPC codes of column weight four and $\mathscr{L}_q^3$–TD LDPC codes of column weight five, respectively, for every prime order $q \leq 100$. Hence, the design strategies admit a wide range of possible codes.

4.7. Simulations

This section demonstrates the decoding power of our well-designed TD LDPC codes based on cyclic-structured transversal designs. Fig. 4.2 shows the bit erasure rates of various LDPC codes over the BEC channel under peeling (or erasure) decoding [53] and Fig. 4.3 shows the bit error rates over the AWGN channel by employing the standard SPA decoder with a maximum of 50 iterations per codeword. A legend displays the following information in the respective order: code type, construction method in brackets, and a triple $[N, R, k]$, where N is the code length, R the code rate and k the constant column weight of the parity-check matrix.

Tabelle 4.11. – Possible parameters of $\mathscr{L}_q^2$–TD LDPC codes of column weight four, length N and rate R. While $\alpha_1 = 1$, the design parameter α_2 is chosen in accordance with the design strategies presented in Section 4.6. The order q is restricted to prime numbers up to 100. Observe that there are no suitable codes for the prime orders $q = 5, 7, 11, 13, 17, 19, 23, 29, 31, 41$.

q	Let $\alpha_1 = 1$. Then, possible choices for α_2 are	N	R
37	5, 8, 9, 14, 15, 17, 21, 23, 24, 29, 30, 33	1369	0.89
43	5, 9, 10, 12, 13, 16, 18, 19, 20, 24, 25, 26, 28, 31, 32, 34, 35, 39	1849	0.91
47	5, 10, 11, 13, 14, 15, 19, 22, 26, 29, 30, 33, 37, 38	2209	0.92
53	5, 6, 7, 8, 9, 10, 11, 13, 15, 16, 17, 20, 21, 22, 25, 29, 32, 33, 34, 37, 38, 39, 41, 43, 44, 45, 46, 47, 48, 49	2809	0.93
59	5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 22, 23, 24, 28, 32, 36, 37, 38, 41, 42, 43, 44, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55	3481	0.93
61	7, 8, 9, 10, 22, 23, 24, 25, 26, 27, 28, 29, 34, 35, 36, 38, 39, 40, 52, 53, 54, 55	3721	0.94
67	5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 18, 19, 20, 21, 24, 25, 26, 27, 28, 32, 36, 40, 41, 42, 43, 44, 47, 48, 49, 53, 54, 56, 57, 59, 60, 61, 62	4489	0.94
71	5, 15, 16, 17, 19, 20, 21, 22, 23, 26, 27, 28, 29, 30, 31, 32, 33, 34, 38, 39, 40, 41, 42, 43, 44, 45, 46, 49, 50, 51, 52, 53, 55, 56, 57, 67	5041	0.94
73	5, 6, 7, 11, 12, 13, 15, 18, 20, 21, 26, 29, 30, 35, 39, 44, 45, 48, 51, 53, 56, 59, 61, 62, 63, 67, 68, 69	5329	0.945
79	5, 6, 7, 12, 13, 14, 15, 16, 17, 18, 21, 22, 28, 32, 33, 34, 37, 38, 42, 43, 46, 47, 48, 52, 58, 59, 62, 63, 64, 65, 66, 67, 68, 73, 74, 75	6241	0.95
83	5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 22, 23, 24, 25, 26, 27, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 57, 58, 59, 60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79	6889	0.95
89	5, 6, 7, 8, 12, 13, 15, 16, 17, 18, 19, 20, 21, 22, 29, 36, 37, 38, 39, 40, 41, 42, 43, 47, 48, 49, 50, 51, 52, 53, 54, 61, 68, 69, 71, 72, 73, 74, 75, 76, 77, 78, 82, 83, 84, 85	7921	0.96
97	5, 6, 9, 10, 11, 12, 17, 18, 19, 20, 24, 26, 27, 28, 29, 30, 31, 34, 39, 40, 41, 42, 43, 44, 45, 46, 47, 51, 52, 53, 54, 55, 56, 57, 58, 59, 64, 67, 68, 69, 70, 71, 72, 74, 78, 79, 80, 81, 86, 87, 88, 89, 92, 93	9409	0.96

Tabelle 4.12. – Possible parameters of $\mathscr{L}_q^3$–TD LDPC codes of column weight five, length N and rate R. The order q is restricted to prime numbers up to 100. While $\alpha_1 = 1$, the design parameters α_2 and α_3 are chosen in accordance with Section 4.6. Observe that there are no suitable codes for the prime numbers $q < 67$.

q	Let $\alpha_1 = 1$. Then, possible choices for (α_2, α_3) are	N	R
67	$(3,15),(3,20),(3,27),(3,37),(5,7),(5,15),(5,39),(5,45)$ $(7,33),(7,53),(7,55),(9,11),(9,13),(9,27),(9,45),$ $(11,23),(11,31),(11,50),(13,33),(13,41),(13,61),\ldots,(61,63)$	4489	0.93
73	$(3,11),(3,15),(3,22),(3,30),(3,40),(3,44),(3,53),(3,60),$ $(4,9),(4,21),(4,28),(4,31),(4,41),(4,45),(4,52),(4,59),$ $(5,7),(5,8),(5,15),(5,28),(5,32),(5,36),(5,49),(5,63),\ldots,(67,69)$	5329	0.93
79	$(7,11),(7,15),(7,32),(7,57),(9,19),(9,32),(9,62),(9,67),$ $(11,23),(11,52),(11,58),(13,28),(13,55),(13,62),(13,71),$ $(15,25),(15,36),(15,48),\ldots,(69,73)$	6241	0.94
83	$(3,8),(3,11),(3,34),(3,38),(3,54),(3,60),(3,66),(3,73),$ $(4,19),(4,38),(4,57),(4,70),(5,8),(5,11),(5,24),(5,36),$ $(5,41),(5,59),(5,67),(5,73),\ldots,(76,81)$	6889	0.94
89	$(3,8),(3,22),(3,56),(3,77),(5,8),(5,13),(5,34),(5,35),(5,48),$ $(5,51),(5,52),(5,60),(5,62),(5,64),(5,65),(5,71),(6,26),$ $(6,52),(6,55),(6,72),(7,18),(7,27),(7,35),(7,53),\ldots,(82,87)$	7921	0.94
97	$(3,8),(3,13),(3,16),(3,17),(3,23),(3,27),(3,35),(3,38),(3,43),$ $(3,45),(3,54),(3,55),(3,56),(3,58),(3,61),(3,69),(3,70),(3,72),$ $(3,74),(3,76),(3,78),(3,79),(3,80),(3,82),(3,83),$ $(3,86),(3,90),(3,93),\ldots,(90,95)$	9409	0.95

Further experimental results have been presented in [P2, Section VII] for the BEC and in [P3, Section VII] for the AWGN channel. In both manuscripts, we have verified that the properly designed TD LDPC codes exhibit a significant improvement of the decoding performance compared to the corresponding TD LDPC codes with an adverse choice of scale factors. Furthermore, the well-chosen codes outperform their counterparts based on the progressive-edge growth (PEG) algorithm [95] and other structured LDPC codes.

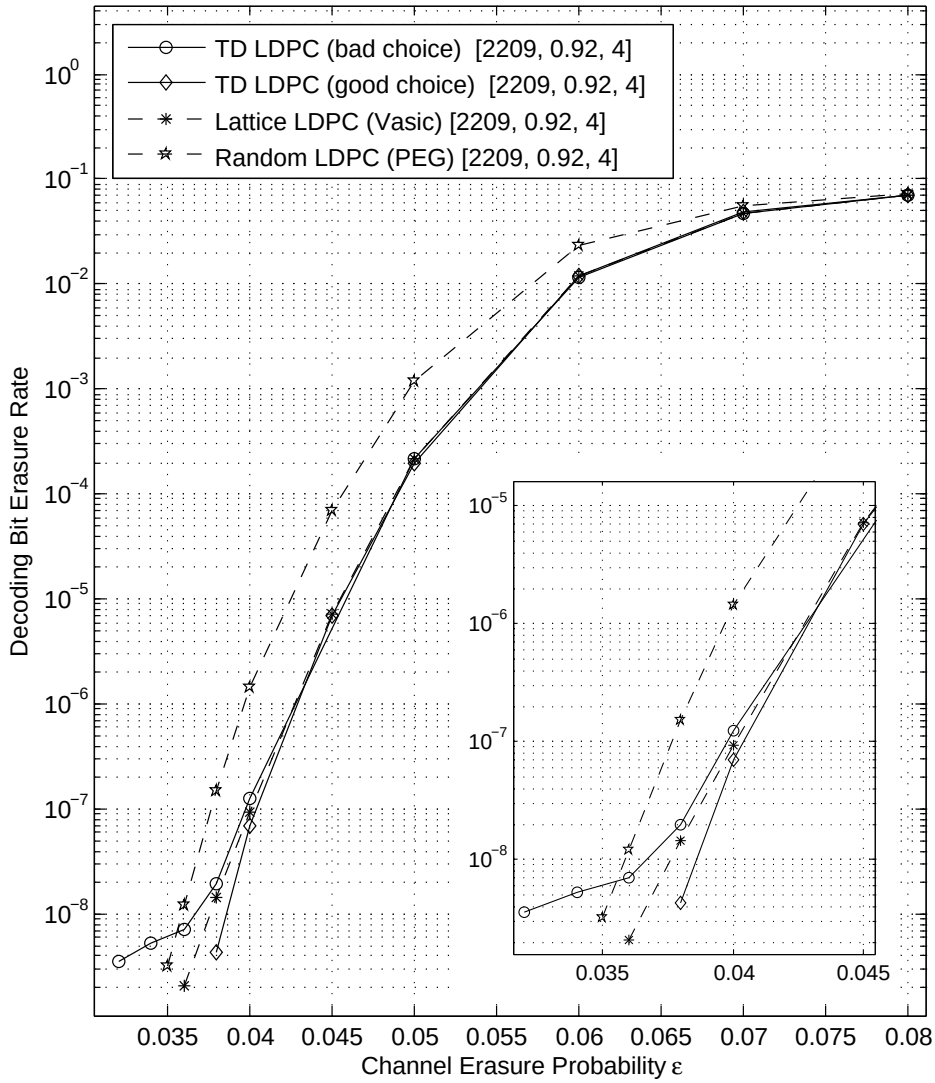


Abbildung 4.2. – The figure compares the BEC decoding performances of an $\mathscr{L}_{47}^2$–TD LDPC code with scale factors $(\alpha_1, \alpha_2) = (1, 2)$ (bad choice) and an $\mathscr{L}_{47}^2$–TD LDPC code with scale factors $(\alpha_1, \alpha_2) = (1, 5)$ (good choice), to a random LDPC code based on the PEG algorithm and to an LDPC code based on the Lattice construction from [35]. All codes have the same code length $N = 2209$, rate $R = 0.92$ and column weight four. It can be observed that the TD LDPC code with proper chosen scale factors outperforms the random LDPC code and the Lattice LDPC code significantly. Also, it can be seen that the TD LDPC code with detrimental scale factors reveals a high error-floor at approximately $\epsilon = 0.04$.

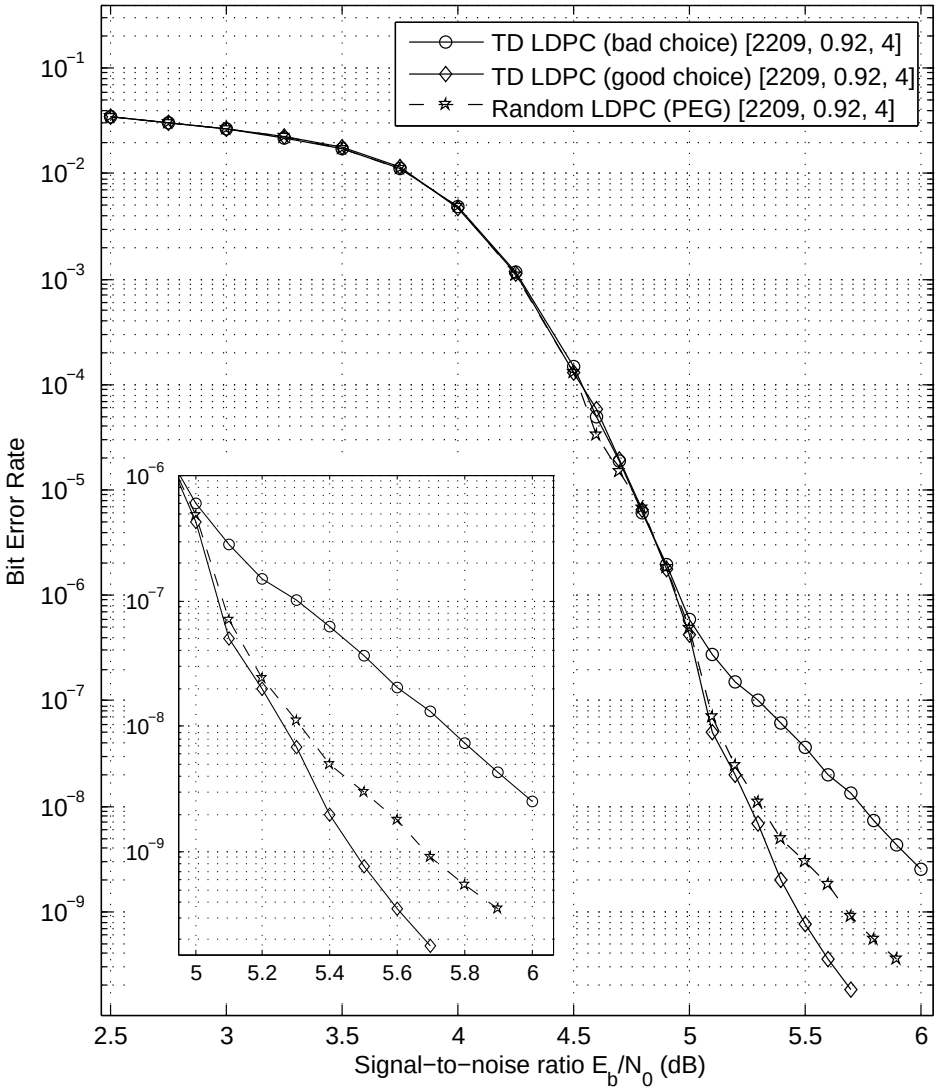


Abbildung 4.3. – The figure compares the AWGN channel decoding performances of an $\mathscr{L}_{47}^2$–TD LDPC code with scale factors $(\alpha_1, \alpha_2) = (1, 2)$ (bad choice) and an $\mathscr{L}_{47}^2$–TD LDPC code with scale factors $(\alpha_1, \alpha_2) = (1, 5)$ (good choice), to a random LDPC code based on the PEG algorithm. All codes have the same code length $N = 2209$, rate $R = 0.92$ and column weight four. It can be observed that the TD LDPC code with well-chosen scale factors outperforms the random LDPC code significantly. By contrast, the adverse TD LDPC code performs very poor and reveals a very high error-floor at approximately 5 dB.

4.8. Discussion

LDPC codes based on BIBDs as presented in the previous chapter have the highest design rates R_d among all 4-cycle-free LDPC codes. Hence, these codes are ideally suited for high-speed data transmission. However, their high combinatorial requirements lead to a rather rigid arrangement of their parity-check matrices, leaving only little scope for code optimizations with respect to their decoding performances. This observation motivates the family of LDPC codes based on transversal designs which allow a more flexible configuration of their parity-check matrices at the cost of a slight decrease of their design rates. Hence, the codes of TD LDPC codes can potentially be more optimized with respect to their decoding performances due to less structural requirements, but are also able to achieve code rates nearly as high as those for BIBD LDPC codes. First, we will take a closer look at the design rates of both code classes:

Proposition 17 *The design rate of BIBD LDPC codes is given by*

$$R_d = 1 - \frac{k(k-1)}{M-1} \tag{4.5}$$

where M is the number of parity-check equations and k is the column weight.

Proof. With $R_d = \frac{N-M}{N}$ and $N = \frac{M(M-1)}{k(k-1)}$, the proposition follows. ■

Proposition 18 *The design rate of TD LDPC codes is given by*

$$R_d = 1 - \frac{k^2}{M} \tag{4.6}$$

where M is the number of parity-check equations and k is the column weight.

Proof. With $R_d = \frac{N-M}{N}$ and $N = \frac{M^2}{k^2}$, the proposition follows. ■

Fig. 4.4 visualizes the difference between the design rates of BIBD and TD LDPC codes for column weights varying from three to ten. As we can observe, there is only a small difference between the design rate curves of BIBD LDPC codes and TD LDPC

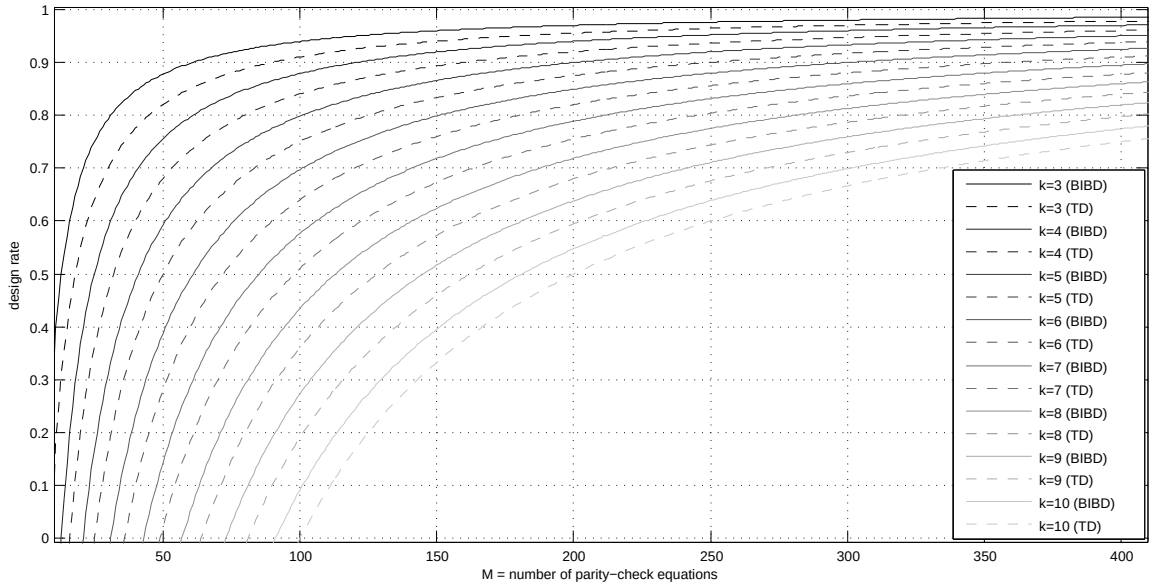


Abbildung 4.4. – Design rates of BIBD LDPC codes vs. TD LDPC codes for column weights 3 to 10

codes (for the same column weight) and, secondly, this difference decreases for growing M. In fact, it can easily be shown that the difference of (4.5) and (4.6) goes to zero if M goes to infinity. Hence, the design rates of the TD LDPC codes are indeed almost as high as those of BIBD LDPC codes.

In the following considerations, we will restrict the class of BIBDs to the subclass of Steiner 2-designs, since the BIBD LDPC codes resulting from Steiner 2-designs naturally avoid 4-cycles in their factor graphs whereas all other BIBD LDPC codes does not share this important property in general. As already mentioned, the small decrease of the design rate of a TD LDPC code compared to a BIBD LDPC code relaxes the combinatorial constraints imposed by the underlying designs. More precisely, every pair of points of a Steiner 2-design must be contained in exactly one block, whereas, by contrast, every pair of points of a TD must only be contained in at most one block of the design. This loosening can potentially lead to a more subtle configuration of the arising parity-check matrix in order to avoid harmful substructures such as stopping

sets and absorbing sets. To shed light on this, the stopping distances of several classes of LDPC codes based on Steiner 2-designs and TDs will be compared in more detail.

As already elaborated in Thm. 2 and 3, the $\mathscr{L}_q^1$-TD and $\mathscr{L}_q^2$-TD LDPC codes of column weight three and four can be designed in such a way that the stopping distances are six and ten, respectively. Furthermore, these codes have enough flexibility to reduce the number of stopping sets of size ten and larger significantly. By contrast, a general lower bound for the stopping distance of LDPC codes based on Steiner 2-designs is stated in [50] by $s_{min} \geq k + 1$. As we can see, the properly designed $\mathscr{L}_q^m$-TD LDPC codes exceed this bound substantially. When we restrict ourselves to the case of quasi-cyclic BIBD LDPC codes of column weight k, an upper bound for the minimum distance has been established in [35, Thm. 3.1] and is given by $d_{min} \leq 2k$ which directly leads to $s_{min} \leq 2k$. In particular, for the case of $k = 4$, we have an upper bound of $s_{min} \leq 8$, whereas, by contrast, the optimized quasi-cyclic $\mathscr{L}_q^2$-TD LDPC codes of column weight four have a higher stopping distance of ten. It is likely that this difference between the stopping distances becomes even greater for growing column weights.

Much attention has been given to the class of Steiner triple systems (STSs) and their corresponding STS LDPC codes of column weight three (e.g. [27, 35, 33]), in particular, to those that avoid so-called Pasch configurations. It has been realized that Pasch configurations in STSs are extremly harmful for the resulting codes, since they correspond to stopping sets of size four [50, 108, 35] and, even more, that these stopping sets are the only possible ones of this size. Hence, the elimination of Pasch configurations in STSs leads to codes with stopping distance $s_{min} \geq 5$ [27]. The STSs that are free of Pasch configurations are termed anti-Pasch STSs and have intensively been studied in the literature independent of any coding theoretic purpose (e.g. [102, 109]). Anti-Pasch STSs are known to be very rare. For instance, from the 80 non-isomorphic STSs of order 15 [110, 111], only one is Pasch-free [102]. Hence, most of the STSs have Pasch configurations and thus result in codes with stopping distance four.

The complete spectrum of orders for which anti-pasch STSs exist is exactly known and given by $\upsilon \equiv 1, 3 \pmod 6$ except for $\upsilon = 7, 13$ [109]. We can observe a loss of parametric flexibility compared to $\mathscr{L}_q^1$-TD LDPC codes that exist for every prime power order q and can trivially avoid Pasch configurations for $\omega_q \neq 2$ due to their specific structure. The situation is being further exacerbated when we restrict ourselves to anti-Pasch STSs that are cyclic. From the 11.084.874.829 STSs of order 19 reported in [112], only two are cyclic and anti-Pasch [113]. By contrast, nearly all $\mathscr{L}_q^1$-TD LDPC codes are quasi-cyclic and thus can be encoded with low complexity.

Further research on STSs in order to avoid certain configurations is concerned with r-sparse STSs. An STS is called *r-sparse* if it does not contain an Erdős configuration (cf. [108]). It can easily be verified that a STS is 4-sparse exactly when it is anti-Pasch [108]. Furthermore, an STS is known to be 5-sparse if it contains no configurations isomorphic to the Pasch configuration or the Mitre configuration (cf. [114]). These 5-sparse STSs are very attractive from a coding theoretic perspective, since they lead to STS LDPC codes with a stopping distance of at least six [108]. Unfortunately, 5-sparse STSs are extremly rare. For instance, from the 2.353.310 cyclic STSs of order 57 listed in [114, Table 1], only 843 are 5-sparse. Hence, there is a significant loss of parametric flexibility compared to $\mathscr{L}_q^1$-TD LDPC codes which exist for every prime power order q and can trivially avoid Pasch and Mitre configurations when $\omega_q \neq 2$. The avoidance of Mitre configurations can easily be verified by the fact that this configuration is not 4-colorable which is a necessary condition in order to occur in any $\mathscr{L}_q^1$-TD LDPC code. The same argument holds for many other stopping set candidates that, by contrast, can potentially occur in STS LDPC codes.

For higher column weights, i.e., when $k \geq 4$, only few is known about the stopping sets of LDPC codes based on Steiner 2-designs. Some studies [35, 50] are concerned with Steiner 2-designs free of *generalized Pasch configurations (GPCs)* which correspond to the smallest possible stopping sets of size $k + 1$ in the LDPC codes based on these designs. Since no other stopping sets of size $k + 1$ can exist, the stopping distances of

these codes are larger than $k+1$ if and only if there is no GPC in the underlying design. Unfortunately, Steiner 2-designs without generalized Pasch-configurations are extremly rare and only few constructions for such designs are known. Possible examples are designs from projective and affine geometries (Chapter 5). More precisely, a projective geometry of prime power order q leads to a Steiner 2-design of block size $q+1$ that avoids GPCs if and only if q is odd and thus to an LDPC code of column weight $q+1$ without stopping sets of size $q+2$. As an example, we obtain an LDPC code of column weight four and stopping distance equal to six (cf. Fig 5.3) based on a projective geometry with $q=3$. By contrast, we have designed TD LDPC codes of column weight four with stopping distances equal to ten which is significantly higher. Furthermore, an affine geometry of prime power order q leads to a Steiner 2-design of block size q avoiding GPCs if and only if q is odd. For instance, we obtain an LDPC code of column weight five based on an affine geometry with $q=5$ and stopping distance of at least seven and at most nine (cf. Fig 5.3). This stopping distance is already exceeded by properly designed TD LDPC codes of column weight four and is expected to be higher for increasing column weights.

As a summary, we can say that for the case of column weight three, there exist some rare 5-sparse STS LDPC codes that achieve the stopping distance of $\mathscr{L}_q^1$-TD LDPC codes. However, the situation changes dramatically for higher column weights, in particular, when we restrict ourselves to quasi-cyclic codes. For example, LDPC codes of column weight four that are based on cyclic Steiner 2-designs have a stopping distance of at most eight [113], whereas $\mathscr{L}_q^2$-TD LDPC codes of column weight four can achieve a stopping distance equal to ten. It is highly probable that this difference between the stopping distances becomes even greater for growing column weights. Hence, it can be expected that the error-floor performance of well designed TD LDPC codes is better than the performance of comparable LDPC codes based on Steiner 2-designs, but at the cost of a slight decrease of the code rates. Although no studies have been found that investigate the absorbing set spectra of LDPC codes based on Steiner 2-designs, it is likely that optimized TD LDPC codes are also more beneficial since they can be designed in such a way that they avoid the most harmful absorbing sets.

5

LDPC Codes Based on Finite Geometries

Low-density parity-check codes based on *finite geometries* (FGs), called *FG LDPC codes*, have originally been proposed in [31, 32] and were among the first algebraic constructed LDPC codes. Subsequently, these codes have been studied in a large series of publications (e.g. [115, 116, 117, 118, 119]) and books (e.g. [120]). This chapter is concerned with the investigation of FG LDPC codes based on certain subclasses of finite geometries, namely projective and affine geometries, and are consequently referred to as *PG LDPC codes* in the case of projective geometries and *AG LDPC codes* in the case of affine geometries. Such codes are well structured and allow efficient encoding and decoding algorithms with low complexity. In particular, they have a quasi-cyclic code representation [32] and thus can be encoded with complexity linear in the code length. Futhermore, it is known that the factors graphs of these codes are free of harmful 4-cycles and have a girth of six.

The main objective of the present chapter is to investigate the stopping distances of FG LDPC codes in order to identify those codes with beneficial stopping set distributions and thus with advantageous error-correcting capabilities over the BEC. The presented results arise from the latest findings in the fields of finite geometries and combinatorial designs and are stated from a coding theoretic perspective. As far as the author know, the derived bounds for the stopping distances are currently the best known.

5.1. Preliminaries

5.1.1. Projective geometry

A *projective geometry* of dimension $m \geq 2$ and prime power order q, denoted by $\mathrm{PG}(m, q)$, is a pair $(\mathcal{P}, \mathcal{L})$ consisting of a point set $\mathcal{P}$ and a line set $\mathcal{L}$ over $\mathbb{F}_q$ such that the points $\mathcal{P}$ are the 1-dimensional subspaces of the vector space $\mathbb{F}_q^{m+1}$, excluding the zero vector, and the lines $\mathcal{L}$ are the 2-dimensional subspaces of $\mathbb{F}_q^{m+1}$, excluding the zero vector. It is well known that

$$|\mathcal{P}| = \frac{q^{m+1}-1}{q-1} \text{ and } |\mathcal{L}| = \frac{(q^{m+1}-1)(q^m-1)}{(q^2-1)(q-1)}.$$

Each line contains $k := q+1$ points and each point lies on $r := (q^m-1)/(q-1)$ lines. If $m = 2$, the $\mathrm{PG}(2, q)$ is called a *projective plane*. Every $\mathrm{PG}(m, q)$ can be described by a binary $|\mathcal{P}| \times |\mathcal{L}|$ incidence matrix, denoted by $\mathcal{N}$, with rows indexed by the points of $\mathcal{P}$, columns indexed by the lines of $\mathcal{L}$ and where $\mathcal{N}_{ij} = 1$ if the i-th point lies on the j-th line, and $\mathcal{N}_{ij} = 0$ otherwise. Furthermore, every $\mathrm{PG}(m, q)$ forms a $\mathrm{BIBD}(\frac{q^{m+1}-1}{q-1}, q+1, 1)$ by associating the points and lines of the projective geometry with the points and blocks of the desin, respectively. In particular, the arising BIBD is a Steiner 2-design.

Let $(\mathcal{P}, \mathcal{L})$ be any $\mathrm{PG}(m, q)$. Each 3-dimensional subspace of $\mathbb{F}_q^{m+1}$ gives the point set $\mathcal{P}'$ of a projective subplane $\mathrm{PG}(2, q)$ with $\mathcal{P}' \subset \mathcal{P}$ and with line set $\mathcal{L}' \subset \mathcal{L}$ which consists of all lines of $\mathcal{L}$ that are completely contained in $\mathcal{P}'$. Consequently, there are many projective subplanes $\mathrm{PG}(2, q)$ embedded in a $\mathrm{PG}(m, q)$.

5.1.2. Affine geometry

An *affine geometry* of dimension $m \geq 2$ and prime power order q, denoted by $\mathrm{AG}(m, q)$, is a pair $(\mathcal{P}, \mathcal{L})$ consisting of a point set $\mathcal{P}$ and a line set $\mathcal{L}$ over $\mathbb{F}_q$ such that the points $\mathcal{P}$ are the vectors of the vector space $\mathbb{F}_q^m$, and the lines $\mathcal{L}$ are the 1-dimensional subspaces

of $\mathbb{F}_q^m$ and their cosets. We have

$$|\mathcal{P}| = q^m \text{ and } |\mathcal{L}| = q^{m-1}\frac{q^m-1}{q-1}.$$

Each line contains $k := q$ points and each point lies on $r := \frac{q^m-1}{q-1}$ lines. If $m = 2$, the $\mathrm{AG}(2, q)$ is called an *affine plane*. Every $\mathrm{AG}(m, q)$ can be described by a binary $|\mathcal{P}| \times |\mathcal{L}|$ *incidence matrix*, $\mathcal{N}$, with rows indexed by the points of $\mathcal{P}$, columns indexed by the lines of $\mathcal{L}$ and with $\mathcal{N}_{ij} = 1$ if the i-th point lies on the j-th line, else $\mathcal{N}_{ij} = 0$.

Every $\mathrm{AG}(m, q)$ forms a $\mathrm{BIBD}(q^m, q, 1)$ by associating the points and lines of the affine geometry with the points and blocks of the design, respectively. In particular, it is a Steiner 2-design. Furthermore, an affine plane $\mathrm{AG}(2, q)$ can be constructed from a projective plane $\mathrm{PG}(2, q)$ by deleting an arbitrary line of $\mathrm{PG}(2, q)$ and all the points lying on this line. All affine planes obtained by deleting any line of $\mathrm{PG}(2, q)$ are isomorphic.

Each coset of a 2-dimensional subspace of $\mathrm{AG}(m, q)$ can be considered as the point set $\mathcal{P}'$ of an affine subplane $\mathrm{AG}(2, q)$. More precisely, let p, p' be two linearly independent points of $\mathbb{F}_q^m \setminus \{(0, \ldots, 0)\}$. Then, the point set can be constructed by $\mathcal{P}' = \{\alpha p + \beta p' + z : \alpha, \beta \in \mathbb{F}_q\}$ for any $z \in \mathbb{F}_q^m$. Finally, the line set $\mathcal{L}'$ of the subplane consists of all lines of $\mathrm{AG}(m, q)$ whose points are completely contained in $\mathcal{P}'$.

5.2. Types of LDPC codes based on finite geometries

By using the incidence matrices of finite geometries as the parity-check matrices of linear block codes, we obtain the family of FG LDPC codes. Based on specific classes of finite geometries, this code family can further be subdivided into several subfamilies. Let $\mathcal{N}$ be the incidence matrix of a $\mathrm{PG}(m, q)$. Then, two types of PG LDPC codes can be derived. First, the *type-I PG LDPC code*, denoted by $C_{\mathrm{PG}}^{(1)}(m, q)$, is given by the nullspace of the (line-by-point) parity-check matrix $H_{\mathrm{PG}}^{(1)}(m, q) := \mathcal{N}^T$ which is the transpose of $\mathcal{N}$.

Second, the *type-II PG LDPC code*, denoted by $C^{(2)}_{\mathrm{PG}}(m,q)$, is given by the nullspace of the (point-by-line) parity-check matrix $H^{(2)}_{\mathrm{PG}}(m,q) := \mathcal{N}$. Analogously, we obtain two types of AG LDPC codes based on the incidence matrix $\mathcal{N}$ of any affine geometry. First, the *type-I AG-LDPC code*, denoted by $C^{(1)}_{\mathrm{AG}}(m,q)$, is given by the parity-check matrix $H^{(1)}_{\mathrm{AG}}(m,q) := \mathcal{N}^T$. Second, the *type-II AG-LDPC code*, denoted by $C^{(2)}_{\mathrm{AG}}(m,q)$, is given by the parity-check matrix $H^{(2)}_{\mathrm{AG}}(m,q) := \mathcal{N}$. This classification has been introduced in [32] and was maintained in nearly all subsequent papers on codes from finite geometries (e.g., [121, 122]).

5.2.1. Properties of FG LDPC codes

An extensive list of fundamental parameters for classical FG LDPC has been summarized and presented in [122]. For the convenience of the reader, some of these results are outlined in Table 5.1. The listed parameters include the code length N, the code dimension K, the column weight k and row weight r of the parity-check matrix, and the minimum distance d_{min} of the code. For calculating the code dimension, which is determined by the 2-rank of the parity-check matrix (or, equivalently, by the incidence matrix of the underlying finite geometry), the following known results are needed.

Theorem 19 [123] *The 2-rank of the incidence matrix of* $\mathrm{PG}(m,2^t)$ *is given by*

$$rank_2(\mathcal{N}) = \varphi(m,2^t) = \sum_{(s_0,\ldots,s_t)} \prod_{j=0}^{t-1} \sum_{i=0}^{L(s_{j+1},s_j)} (-1)^i \binom{m+1}{i} \binom{m+2s_{j+1}-s_j-2i}{m}$$

where the first sum is taken over all ordered sets $(s_0,\ldots,s_t)$ *with* $s_0 = s_t,\ s_j \in \mathbb{Z}$ *such that* $0 \le s_j \le m-1$ *and* $0 \le 2s_{j+1} - s_j \le m+1$ *for each* $j = 0,\ldots,t-1$, *and*

$$L(s_{j+1},s_j) = \left\lfloor \frac{2s_{j+1}-s_j}{2} \right\rfloor.$$

Theorem 20 [124] *For* q *odd, the 2-rank of the incidence matrix of* $\mathrm{PG}(m,q)$ *is*

$$rank_2(\mathcal{N}) = |\mathcal{P}| - 1 = \frac{q^{m+1}-q}{q-1}.$$

FG	Type	m	q	N	K	(k,r)	d_{min}
PG	I	any	2^t	$\frac{2^{t(m+1)}-1}{2^t-1}$	$N-\varphi(m,2^t)$	$\left(\frac{2^{tm}-1}{2^t-1}, 2^t+1\right)$	$(2^t+2)2^{t(m-2)}$
PG	II	any	2^t	$\frac{(2^{t(m+1)}-1)(2^{tm}-1)}{(2^{2t}-1)(q-1)}$	$N-\varphi(m,2^t)$	$\left(2^t+1, \frac{2^{tm}-1}{2^t-1}\right)$	2^t+2
PG	II	any	odd	$\frac{(q^{m+1}-1)(q^m-1)}{(q^2-1)(q-1)}$	$N-\frac{q^{m+1}-q}{q-1}$	$\left(q+1, \frac{q^m-1}{q-1}\right)$	$2(q+1)$
AG	I	any	2^t	2^{tm}	$N-\varphi(m,2^t)+\varphi(m-1,2^t)$	$\left(\frac{2^{tm}-1}{2^t-1}, 2^t\right)$	$(2^t+2)2^{t(m-2)}$
AG	II	any	2^t	$2^{t(m-1)}\frac{2^{tm}-1}{2^t-1}$	$N-\varphi(m,2^t)+\varphi(m-1,2^t)$	$\left(2^t, \frac{2^{tm}-1}{2^t-1}\right)$	2^t+1
AG	II	any	odd	$q^{m-1}\frac{q^m-1}{q-1}$	$N-q^m$	$\left(q, \frac{q^m-1}{q-1}\right)$	$2q$

Abbildung 5.1. – Parameters of FG LDPC codes, where q is generally restricted to be a prime power and $m \geq 2$ (cf. [122]). The function $\varphi(m, 2^t)$ is given by Thm. 19.

Theorem 21 [125] *The 2-rank of the incidence matrix of* $\mathrm{AG}(m, 2^t)$ *is given by*

$$rank_2(\mathcal{N}) = \varphi(m, 2^t) - \varphi(m-1, 2^t).$$

Theorem 22 [126] *For q odd, the 2-rank of the incidence matrix of* $\mathrm{AG}(m, q)$ *is*

$$rank_2(\mathcal{N}) = |\mathcal{P}| = q^m.$$

The code bits of type-I FG LDPC codes are associated with the points $\mathcal{P}$ of the finite geometry such that $N = |\mathcal{P}|$, and the parity-check equations are associated with the lines of the geometry, i.e., $M = |\mathcal{L}|$. For an odd order q, the code dimension is given by $K = |\mathcal{P}| - \mathrm{rank}_2(H^{(1)}_{\mathrm{PG}}(m,q)) = 1$ (Thm. 20) in the case of type-I PG LDPC codes and by $K = |\mathcal{P}| - \mathrm{rank}_2(H^{(1)}_{\mathrm{AG}}(m,q)) = 0$ (Thm. 22) in the case of type-I AG LDPC codes. As a consequence, the type-I FG LDPC codes are useless for practical application when q is odd. For type-I FG LDPC codes, we therefore restrict our considerations to the case of $q = 2^t$ even, leading to finite geometries which are conjectured in [125] to have the lowest rank among all Steiner 2-designs of the same order and block size (cf. [122]). The resulting type-I FG LDPC codes therefore achieve the highest possible code rates as t increases.

The code bits of type-II FG LDPC codes are associated with the lines $\mathcal{L}$ of the finite geometry such that $N = |\mathcal{L}|$, and the parity-check equations are associated with the points, i.e., $M = |\mathcal{P}|$. For the case of $q = 2^t$, these codes are known to have the lo-

west possible minimum distance [122] and thus may perform suboptimally over various channels. By contrast, if q odd, the type-II FG-LDPC codes have very high minimum distances [122], even achieving the highest possible minimum distances among all codes based on non-trivial abelian point-transitive Steiner 2-designs of the same order and block size [113].

5.3. Stopping set analysis

In this section, we present several bounds for the stopping distances of LDPC codes based on projective and affine geometries. These bounds are mainly derived from the latest theoretical findings in the field of finite geometries that have been studied independently of any coding theoretic purpose. It is worth noting here that these bounds are valid for every single code within the family of LDPC codes based on projective and affine geometries and thus are of great significance.

5.3.1. Stopping distance of PG LDPC codes

First, we consider the case of LDPC codes based on projective planes $\mathrm{PG}(2,q)$. In this case, the concept of stopping sets can be equivalently transferred to the level of the underlying projective planes, leading to the concept of *sets without tangents*. These combinatorial entities have been intensively studied in the field of finite geometries (cf. [127, 128] and the references therein) and thus are a valuable source for our investigations. More specifically, let $\mathcal{S}$ be any subset of the points of a projective plane $\mathrm{PG}(2,q)$. A line of $\mathrm{PG}(2,q)$ that meets $\mathcal{S}$ in exactly one point is called a *tangent* line of $\mathcal{S}$. If there is no tangent line of $\mathcal{S}$, then $\mathcal{S}$ is called a *set without tangents*.

Lemma 23 (cf. [128]) *A set without tangents $\mathcal{S}$ of size $\ell := |\mathcal{S}|$ in a projective plane $\mathrm{PG}(2,q)$ is equivalent to a stopping set of size ℓ in $H^{(1)}_{\mathrm{PG}}(2,q)$ and is the dual representation of a stopping set of size ℓ in $H^{(2)}_{\mathrm{PG}}(2,q)$.*

Proof. Since the columns of $H_{\mathrm{PG}}^{(1)}(2,q)$ correspond to the points of $\mathrm{PG}(2,q)$, the definition of a set without tangents is equivalent to the definition of a stopping set. If we interchange the roles of points and lines of a projective plane π, we obtain a dual plane π^* which is isomorphic to π. This implies that each set without tangents has a dual set of lines which correspond to the columns of a stopping set of $H_{\mathrm{PG}}^{(2)}(2,q)$. ■

Type-I PG LDPC codes

By considering the line-by-point indidence matrix of a $\mathrm{PG}(m,q)$ as a parity-check matrix such that the lines correspond to the parity-check equations and the points correspond to the code bits, we obtain a type-I PG LDPC code. When the order q of $\mathrm{PG}(m,q)$ is odd and $m > 2$, then the arising type-I PG LDPC codes are useless for any practical application since their code rates are close to zero. We therefore restrict our considerations to the case of $q = 2^t$ even.

Proposition 24 *If $q = 2^t$ is even, then*

$$\frac{2^{tm}-1}{2^t-1} + 1 \leq s_{min}(H_{\mathrm{PG}}^{(1)}(m,2^t)) \leq (2^t+2)2^{t(m-2)}.$$

For $m = 2$, it follows that $s_{min}(H_{\mathrm{PG}}^{(1)}(2,2^t)) = 2^t + 2$.

Proof. Recall that the columns of $H_{\mathrm{PG}}^{(1)}(m,2^t)$ correspond to the points of $\mathrm{PG}(m,2^t)$. Since every point lies on $r = \frac{2^{tm}-1}{2^t-1}$ lines, each column has weight r. It can easily be seen that we need at least $r+1$ columns to construct a stopping set, such that the submatrix formed by these columns has no rows of weight one. The lower bound follows. The minimum distance of the code $C_{\mathrm{PG}}^{(1)}(m,2^t)$ is known to be $d_{min}(C_{\mathrm{PG}}^{(1)}(m,2^t)) = (2^t + 2)2^{t(m-2)}$ [129, Thm. 1] which is an upper bound for $s_{min}(H_{\mathrm{PG}}^{(1)}(m,2^t))$. ■

Type-II PG LDPC codes

By taking the point-by-line indidence matrix of a $\mathrm{PG}(m,q)$ as a parity-check matrix such that the points correspond to the parity-check equations and the lines correspond to the

code bits, we obtain a type-II PG LDPC code. Let $\mathrm{PG}(m,q)$ be a projective geometry with point set $\mathcal{P}$ and line set $\mathcal{L}$. Then, we say that a subset of lines $\mathcal{E} \subseteq \mathcal{L}$ is a stopping set of $\mathrm{PG}(m,q)$ when each point of $\mathcal{P}_\mathcal{E} = \bigcup \mathcal{E}$ is contained in at least two lines of $\mathcal{E}$. By saying that $\mathcal{E}$ is a stopping set of $\mathrm{PG}(m,q)$, we mean that the lines of $\mathcal{E}$ correspond to columns of $H_{\mathrm{PG}}^{(2)}(m,q)$ which form a stopping set.

Proposition 25 *For any stopping set $\mathcal{E}$ of $\mathrm{PG}(m,q)$ it holds either that $\mathcal{E}$ occurs in a projective subplane $\mathrm{PG}(2,q)$ or that the size of $\mathcal{E}$ can be lower bounded by $|\mathcal{E}| \geq 2q+2$.*

Proof. See Appendix F.1. ■

Theorem 26 *For any prime power order q, it holds that*

$$s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) = s_{min}(H_{\mathrm{PG}}^{(2)}(2,q)).$$

Proof. Recall that there are many subplanes $\mathrm{PG}(2,q)$ embedded in a $\mathrm{PG}(m,q)$ and thus, every stopping set of $\mathrm{PG}(2,q)$ directly leads to a stopping set of $\mathrm{PG}(m,q)$, giving $s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) \leq s_{min}(H_{\mathrm{PG}}^{(2)}(2,q))$.[1] Conversely, assume that there is a stopping set $\mathcal{E}$ in $\mathrm{PG}(m,q)$ of size $|\mathcal{E}| < s_{min}(H_{\mathrm{PG}}^{(2)}(2,q))$ which implies that $\mathcal{E}$ does not occur in any $\mathrm{PG}(2,q)$. We also know that $s_{min}(H_{\mathrm{PG}}^{(2)}(2,q))$ is upper bounded by $d_{min}(H_{\mathrm{PG}}^{(2)}(2,q)) = 2q+2$ [122, Thm. 27]. It follows that $|\mathcal{E}| < 2q+2$ which is a contradiction to Prop. 25. Hence, $|\mathcal{E}| \geq s_{min}(H_{\mathrm{PG}}^{(2)}(2,q))$ and thus $s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) \geq s_{min}(H_{\mathrm{PG}}^{(2)}(2,q))$. ■

Proposition 27 *If $q = 2^t$ is an even prime power, then*

$$s_{min}(H_{\mathrm{PG}}^{(2)}(m,2^t)) = 2^t + 2.$$

Proof. By considering $\mathrm{PG}(m,2^t)$ as a Steiner 2-design of block size 2^t+1, we know from [50] that the stopping distance is $s_{min}(H_{\mathrm{PG}}^{(2)}(m,2^t)) \geq 2^t+2$. For $q = 2^t$, a stopping set of size 2^t+2 in $H_{\mathrm{PG}}^{(2)}(2,2^t)$ corresponds to the dual of a hyperoval in $\mathrm{PG}(2,2^t)$.

[1] As an example, Fig. 5.2 depicts the embedding of the parity-check matrix $H_{\mathrm{PG}}^{(2)}(2,2)$ in the parity-check matrix $H_{\mathrm{PG}}^{(2)}(3,2)$.

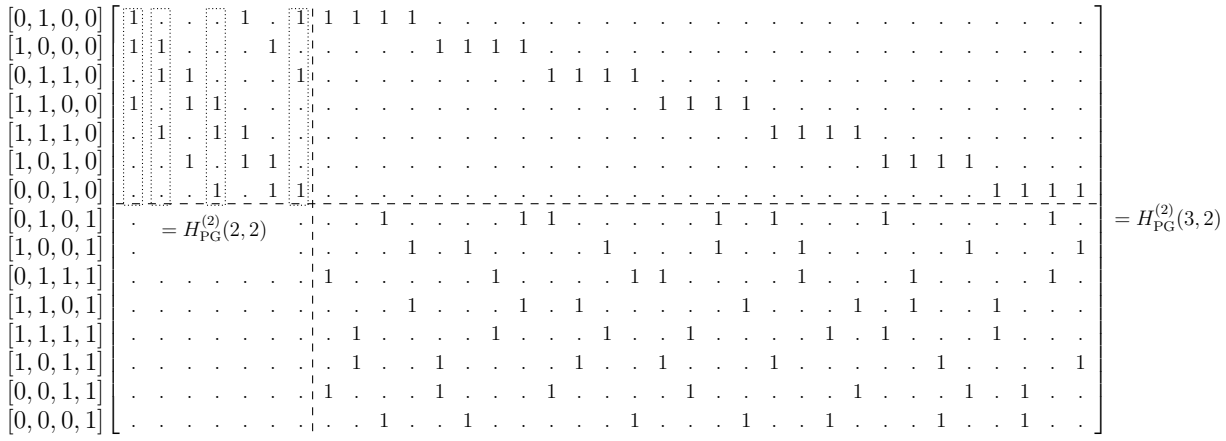


Abbildung 5.2. – Embedding of $H^{(2)}_{\mathrm{PG}}(2,2)$ in $H^{(2)}_{\mathrm{PG}}(3,2)$; The stopping set in $H^{(2)}_{\mathrm{PG}}(2,2)$, formed by the columns 1, 2, 4 and 7, is also a stopping set in $H^{(2)}_{\mathrm{PG}}(3,2)$.

A *hyperoval* $\mathcal{O}$ is a set of $2^t + 2$ points such that any line of $\mathrm{PG}(2, 2^t)$ meets $\mathcal{O}$ in either zero or exactly two points of $\mathcal{O}$. It is well known that a hyperoval exists in $\mathrm{PG}(2, q)$ if and only if q is even [50]. Since a hyperoval is a set without tangents in $\mathrm{PG}(2, 2^t)$, there is a stopping set in $H^{(2)}_{\mathrm{PG}}(2, 2^t)$ of the same size (Lemma 23), such that $s_{min}(H^{(2)}_{\mathrm{PG}}(2, 2^t)) = 2^t + 2$. The generalized proposition follows from Thm. 26. ■

Proposition 28 *For any prime power q, it holds that*

$$s_{min}(H^{(2)}_{\mathrm{PG}}(m, q)) \leq 2q.$$

Proof. Let L, L' be two lines of a projective plane $\mathrm{PG}(2, q)$ with an intersection point p. The points of $(L \cup L') \setminus \{p\}$ form a set without tangents of size $2q$ [128] which corresponds to a stopping set of the same size according to Lemma 23. Hence, $s_{min}(H^{(2)}_{\mathrm{PG}}(2, q)) \leq 2q$. The generalized proposition follows from Thm. 26. ■

Proposition 29 *If q is an odd prime power and $q > 5$, then*

$$s_{min}(H^{(2)}_{\mathrm{PG}}(m, q)) \leq 2(q - 1).$$

Proof. In a projective plane $\mathrm{PG}(2, q)$ with $q > 5$ and point set $\mathcal{P}$, let $\mathcal{C}_1 = \{[x, y, z] \in \mathcal{P} : z^2 = xy\}$ (over $\mathbb{F}_q$) and $\mathcal{C}_2 = \{[x, y, z] \in \mathcal{P} : z^2 = axy\}$ (over $\mathbb{F}_q$) with $a \in \mathbb{F}_q$ such

that $a-1$ and $a(a-1)$ are both squares over $\mathbb{F}_q$. The union $\mathcal{C}_1 \cup \mathcal{C}_2$ is a set without tangents of size $2(q-1)$ [130, 128]. The prop. follows from Lemma 23 and Thm. 26. ■

Proposition 30 *If $q = p^t$ is an odd prime power with p prime and $t \geq 1$, then*

$$s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) \leq 2q + 1 - \frac{q-p}{p-1}.$$

Proof. A *blocking set* of $\mathrm{PG}(2,q)$ is a subset of points such that each line contains at least one point of the blocking set. A blocking set is *minimal*, if no subset is a blocking set. In [131], a minimal blocking set B of size $|B| = q + \frac{q-1}{p-1}$ is constructed by

$$B = \Big\{[1, x, x^p] : x \in \mathbb{F}_{p^t}\Big\} \cup \Big\{[0, x, x^p] : x \in \mathbb{F}_{p^t}, x \neq 0\Big\}.$$

Since B is a blocking set of *Rédei-type* [132], there exists a line L with $q+1$ points such that $|B \cap L| = \frac{q-1}{p-1}$. Then, the set $(B \cup L) \setminus (B \cap L)$ is a set without tangents of size

$$\ell := |B| + |L| - 2|B \cap L| = 2q + 1 - \frac{q-1}{p-1},$$

which corresponds to a stopping set of the same size in $H_{\mathrm{PG}}^{(2)}(2,q)$ (Lemma 23), such that $s_{min}(H_{\mathrm{PG}}^{(2)}(2,q)) \leq \ell$. The generalized proposition follows from Thm. 26. ■

Proposition 31 *If q is an odd prime power, then*

$$s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) \geq \begin{cases} q+5, & \text{if } 3 \nmid q, \\ q+3, & \text{else.} \end{cases}$$

Proof. These bounds has been presented in the full version of [50, Thm. 13 and 14] for the case of projective planes and can be generalized to m-dimensional projective geometries by applying Thm. 26. ■

Proposition 32 *If q is an odd prime power, then*

$$s_{min}(H_{\mathrm{PG}}^{(2)}(m,q)) \geq q + \frac{1}{4}\sqrt{2q} + 2.$$

Proof. This lower bound was introduced by Blokhuis, Seress and Wilbrink in [130] for the size of sets without tangents (which were called *untouchable sets*) in projective planes. The proposition follows with Lemma 23 and Thm. 26. ■

5.3.2. Stopping distance of AG LDPC codes

Type-I AG LDPC codes

We obtain a type-I AG LDPC code by considering the line-by-point incidence matrix of an $\mathrm{AG}(m,q)$ as the parity-check matrix of the code. Then, the lines of the geometry correspond to the parity-check equations and the points correspond to the code bits.

Proposition 33 *If $q = 2^t$ is an even prime power order, then*

$$\frac{2^{tm}-1}{2^t-1} + 1 \leq s_{min}(H_{\mathrm{AG}}^{(1)}(m,2^t)) \leq (2^t+2)2^{t(m-2)}.$$

For $m = 2$, we have $s_{min}(H_{\mathrm{AG}}^{(1)}(2,2^t)) = 2^t + 2$.

Proof. The columns of $H_{\mathrm{PG}}^{(1)}(m,2^t)$ have weight $r = \frac{2^{tm}-1}{2^t-1}$. It can be easily seen that we need at least $r+1$ columns to establish a stopping set, such that the submatrix formed by these columns has no rows of weight one. The lower bound follows. The minimum distance of $C_{\mathrm{AG}}^{(1)}(m,2^t)$ is known to be $d_{min}(C_{\mathrm{AG}}^{(1)}(m,2^t)) = (2^t+2)2^{t(m-2)}$ [129, Thm. 1] which is an upper bound for $s_{min}(H_{\mathrm{AG}}^{(1)}(m,2^t))$. ■

Type-II AG LDPC codes

Let $\mathrm{AG}(m,q)$ be an affine geometry with point set $\mathcal{P}$ and line set $\mathcal{L}$. By taking the point-by-line indidence matrix of $\mathrm{AG}(m,q)$ as the parity-check matrix of an LDPC code such that the points correspond to the parity-check equations and the lines correspond to the code bits, we obtain a type-II AG LDPC code. Then, we say that a subset of lines $\mathcal{E} \subseteq \mathcal{L}$ is a stopping set of $\mathrm{AG}(m,q)$, when each point of $\mathcal{P}_\mathcal{E} = \bigcup \mathcal{E}$ is contained in at least two lines of $\mathcal{E}$. By saying that $\mathcal{E}$ is a stopping set of $\mathrm{AG}(m,q)$, it is meant that there is a corresponding stopping set in the factor graph of $H_{\mathrm{AG}}^{(2)}(m,q)$.

Proposition 34 *For any stopping set $\mathcal{E}$ of $AG(m,q)$ it holds either that $\mathcal{E}$ occurs in an affine subplane $AG(2,q)$ or that the size of $\mathcal{E}$ can be lower bounded by $|\mathcal{E}| \geq 2q$.*

Proof. See Appendix F.2. ■

Theorem 35 *For any prime power order q, it holds that*

$$s_{min}(H^{(2)}_{\mathrm{AG}}(m,q)) = s_{min}(H^{(2)}_{\mathrm{AG}}(2,q)).$$

Proof. Recall that there are many affine subplanes $\mathrm{AG}(2,q)$ embedded in the affine geometry $\mathrm{AG}(m,q)$ and thus every stopping set of $\mathrm{AG}(2,q)$ directly results in a stopping set of $\mathrm{AG}(m,q)$ such that $s_{min}(H^{(2)}_{\mathrm{AG}}(m,q)) \leq s_{min}(H^{(2)}_{\mathrm{AG}}(2,q))$. Now, we assume that there is a stopping set $\mathcal{E}$ in $\mathrm{AG}(m,q)$ of size $|\mathcal{E}| < s_{min}(H^{(2)}_{\mathrm{AG}}(2,q))$ which implies that $\mathcal{E}$ does not occur in any $\mathrm{AG}(2,q)$. We also know that $s_{min}(H^{(2)}_{\mathrm{AG}}(2,q))$ is upper bounded by $d_{min}(H^{(2)}_{\mathrm{AG}}(2,q)) = 2q$ [122, Thm. 34] such that $|\mathcal{E}| < 2q$. This is a contradiction to Prop. 34. Hence, $|\mathcal{E}| \geq s_{min}(H^{(2)}_{\mathrm{AG}}(2,q))$ and thus $s_{min}(H^{(2)}_{\mathrm{AG}}(m,q)) \geq s_{min}(H^{(2)}_{\mathrm{AG}}(2,q))$. ■

Proposition 36

$$s_{min}(H^{(2)}_{\mathrm{AG}}(m,q)) \leq s_{min}(H^{(2)}_{\mathrm{PG}}(m,q)) - 1$$

Proof. Let $\mathcal{E}$ be the set of lines of $\mathrm{PG}(2,q)$ that correspond to a stopping set in $H^{(2)}_{\mathrm{PG}}(2,q)$. After deleting any line $L \in \mathcal{E}$ and all points lying on L, the remaining lines $\mathcal{E} \setminus L$ correspond to a stopping set in $H^{(2)}_{\mathrm{AG}}(2,q)$ of size $|\mathcal{E}| - 1$, where $\mathrm{AG}(2,q)$ arises from $\mathrm{PG}(2,q)$ by deleting L and all points lying on L. Hence, $s_{min}(H^{(2)}_{\mathrm{AG}}(2,q)) \leq s_{min}(H^{(2)}_{\mathrm{PG}}(2,q)) - 1$. The proposition finally follows with Thm. 26 and Thm. 35. ■

Proposition 37 *If $q = 2^t$ is an even prime power, then*

$$s_{min}(H^{(2)}_{\mathrm{AG}}(m,2^t)) = 2^t + 1.$$

Proof. By considering $\mathrm{AG}(m,2^t)$ as a Steiner 2-design with block size 2^t, we know from [50] that the stopping distance must be $s_{min}(H^{(2)}_{\mathrm{AG}}(m,2^t)) \geq 2^t+1$. For $q = 2^t$, there are stopping sets of size $2^t + 2$ in $H^{(2)}_{\mathrm{PG}}(m,2^t)$ (Prop. 27), leading to $s_{min}(H^{(2)}_{\mathrm{AG}}(m,2^t)) \leq 2^t + 1$ with Prop. 36. ■

Proposition 38 *If q is an odd prime power, then*

$$s_{min}(H_{\mathrm{AG}}^{(2)}(m,q)) \leq \begin{cases} 2q-1, & \\ 2q-3, & \text{if } q > 5, \\ 2q-\frac{q-p}{p-1}, & \text{if } q = p^t. \end{cases}$$

Proof. This follows by applying Prop. 36 to the results of Prop. 28, 29 and 30. ■

Proposition 39 *If q is an odd prime power, then*

$$s_{min}(H_{\mathrm{AG}}^{(2)}(m,q)) \geq q+2$$

Proof. This lower bound has been established in [50] for the case of affine planes ($m = 2$). The generalized proposition follows by Thm. 35. ■

5.4. Discussion

The problem of calculating the stopping distance of codes based on finite geometries has been reduced to the problem of calculating the stopping distance of codes based on the corresponding projective and affine planes (cf. Thm. 26 and 35). As a consequence, the derived bounds for codes on finite planes with $m = 2$ can be directly transferred to the general case as m increases. As a summary, Fig. 5.3 gives on overview of the determined bounds for the stopping distances of PG and AG LDPC codes in comparison to their minimum distances. In particular note, that the derived bounds are generally valid for all parameters q and m and thus for all possible codes based on affine and projective geometries.

It can be observed that type-I FG LDPC codes have very large stopping distances which can be explained by large column weights as m increases. Since these codes are highly rank-deficient, they can reach high code rates although they have more parity-check equations than code bits. However, the decoding of such codes is computationally

FG	Type	m	q (prime power)	s_{min}		d_{min}
PG	I	≥ 2	2^t	$\leq (2^t+2)2^{t(m-2)}$	(Prop. 24)	$(2^t+2)2^{t(m-2)}$
PG	I	≥ 2	2^t	$\geq \frac{2^{tm}-1}{2^t-1}+1$	(Prop. 24)	
PG	I	$=2$	2^t	2^t+2	(Prop. 24)	2^t+2
PG	II	≥ 2	2^t	$q+2$	(Prop. 27)	$q+2$
PG	II	≥ 2	odd	$\leq 2q$	(Prop. 28)	$2(q+1)$
PG	II	≥ 2	$q>5$ odd	$\leq 2(q-1)$	(Prop. 29)	
PG	II	≥ 2	$q=p^t$ odd	$\leq 2q+1-\frac{q-p}{p-1}$	(Prop. 30)	
PG	II	≥ 2	odd	$\geq q+3$	(Prop. 31)	
PG	II	≥ 2	odd and $3 \nmid q$	$\geq q+5$	(Prop. 31)	
PG	II	≥ 2	odd	$\geq q+\frac{1}{4}\sqrt{2q}+2$	(Prop. 32)	
AG	I	≥ 2	2^t	$\leq (2^t+2)2^{t(m-2)}$	(Prop. 33)	$(2^t+2)2^{t(m-2)}$
AG	I	≥ 2	2^t	$\geq \frac{2^{tm}-1}{2^t-1}+1$	(Prop. 33)	
AG	I	$=2$	2^t	2^t+2	(Prop. 33)	2^t+2
AG	II	≥ 2	2^t	$q+1$	(Prop. 37)	$q+1$
AG	II	≥ 2	odd	$\leq 2q-1$	(Prop. 38)	$2q$
AG	II	≥ 2	$q>5$ odd	$\leq 2q-3$	(Prop. 38)	
AG	II	≥ 2	$q=p^t$ odd	$\leq 2q-\frac{q-p}{p-1}$	(Prop. 38)	
AG	II	≥ 2	odd	$\geq q+2$	(Prop. 39)	

Abbildung 5.3. – Bounds for the stopping distance s_{min} of FG LDPC codes, where q is generally restricted to be a prime power and $m \geq 2$.

intensive and suboptimal due the large number of linearly dependent parity-check equations. By contrast, type-II FG LDPC codes have nearly full 2-ranks and thus are efficient in that sense that their computational decoding effort is in an optimal relation to their decoding performance. It can be seen, that it is very important to avoid type-II FG LDPC codes with q even, since these codes have extremly low stopping distances as well as minimum distances. On the other hand, type-II FG LDPC codes with q odd have large stopping distances and thus show an excellent decoding performance, in particular, in the error-floor region.

6

Conclusion

Although intensive research in the field of coding theory has been made significant progress in constructing error-correcting codes with capacity-achieving decoding performances, it is still a major task to design well-performing codes along with ease of implementation. The present thesis makes substantial progress in the design and analysis of structured LDPC and sRA codes based on combinatorial designs that can potentially be used for high-speed applications. These algebraic codes show excellent decoding performances and are designed such that they can be encoded with very low complexity.

In Chapter 3, novel families of structured LDPC codes have been presented based on certain subclasses of combinatorial designs, more specifically, based on CBIBDs, RBIBDs and CRCBIBDs. The structure of LDPC codes from CBIBDs have the advantage that they enable low-complexity encoding linear in the code length and the LDPC codes based on RBIBDs reveal a great parametric flexibility since their lengths and rates can be adjusted independently by removing an arbitrary selection of resolution classes. Finally, LDPC codes based on CRCBIBDs combine the strength of both code families. Futhermore, these results have been adapted to construct new families of systematic repeat-accumulate codes which can generally be encoded with linear complexity as opposed to unstructured random-like LDPC codes. For short to moderate code lengths, classical sRA codes have the drawback that they possess columns of weight two in their parity-check matrices which cause a relatively high error-floor. This handicap has been

eliminated by introducing the family of weight-q sRA codes as a generalization of the classical sRA codes. Subsequently, we have designed new weight-q sRA code with excellent decoding performances similar to their LDPC counterparts, but along with the low-encoding complexity of turbo-like codes.

Chapter 4 is concerned with an extensive analysis of the stopping set and absorbing set spectra of LDPC codes based on transversal designs. By utilizing the results of these investigations, powerful design strategies have been derived in order to construct infinite families of high-rate structured LDPC codes with excellent decoding performances over the BEC and the AWGN channel, in particular, with very low error-floors. Besides their good performances in the high-SNR region, LDPC codes from transversal design are quasi-cyclic, resolvable and have basically the same good structural properties as LDPC codes based on BIBDs, for instance, their factor graphs are free of 4-cycles.

Finally, Chapter 5 presents new theoretical results for the stopping distances of LDPC codes based on finite geometries, more precisely, based on projective and affine geometries. Firstly, it has been shown that the problem of determining the stopping distances of codes based on finite geometries can be reduced to the much simpler problem of finding the stopping distances of codes based on the corresponding (projective and affine) planes. Secondly, deep theoretical insights are generated by deriving tight bounds for the stopping distances of such codes based on the latest findings in the fields of finite geometries and combinatorial designs. These bounds are essential to reveal those codes with the most beneficial high-SNR performance. As a result, we obtain structured LDPC codes with excellent decoding performances over the BEC.

6.1. Outlook

There are several possibilites how the present thesis can be followed up. First notice that the combinatorial design theory is an old discipline that goes back several centuries [80] such that the concepts from combinatorial design theory have been investigated far

earlier than the field of coding theory came into existence. Subsequently, the universe of design theory expanded rapidly year by year and is still an ongoing subject of research. As a consequence, many fundamental designs are well investigated and a large variety of constructions for such designs exist and are still developed. Since combinatorial designs can be used for the construction of LDPC codes, the wide range of combinatorial designs has been and is still a valuable source for the construction of novel and structured code families.

Many structured LDPC codes have already been designed based on known families of Steiner 2-designs including finite geometries [32], unitals [76], oval designs [133] and so on. Furthermore, there exist codes based on the more generalized concepts such as partial geometries [36, 27, 83] which include the Steiner 2-designs as a subclass. However, only a small fraction of possible constructions for such designs has been considered and further examined from a coding theoretic perspective. There are still many direct as well as recursive constructions for Steiner 2-designs and more generalized concepts that are yet unconsidered for generating an LDPC code. These constructions potentially lead to novel LDPC codes with completely different structures and decoding performances than existing ones even if their basic parameters are identical. Possible constructions for exploitable Steiner 2-designs can be found, for example, in [134, 102, 113]. For a survey of known results for the existence and construction of Steiner 2-designs see [79, 80, 135, 136] and the references therein.

While a wide range of LDPC codes with column weight three has already been proposed based on Steiner triple systems, there is still a lack of structured LDPC codes with larger column weights, i.e., with $k > 3$. For instance, possible construction techniques for Steiner 2-designs of block size four are described in [137, Section 2], leading to potential new families of LDPC codes. It is also worth to consider the more generalized concepts of *Pairwise Balanced Designs (PBDs)* and *Group Divisible Designs (GGDs)* [79, Part IV] whose incidence structures are further convenient sources for new LDPC codes.

As a next step, the novel and existing families of structured LDPC codes can be

thoroughly examined with regard to the most harmful trapping sets that occur under iterative decoding. While the decoding failure mechanisms over the BEC can completely be explained in terms of stopping sets, the situation is more complicated for non-erasure channels such as the BSC or AWGN channel. For these channels, it is still an open problem how trapping sets can be fully characterized and how they behave under message-passage decoding. Hence, a better characterization of the failure mechanisms would be desirable. An advisable first approach is to identify the most relevant stopping and absorbing sets due to the simple combinatorial characteristic of these entities and also, since absorbing sets are known to be the main cause of error-floors over the AWGN channel. By exploiting the specific structure of the combinatorial code families, this task can be simplified considerably.

Moreover, it would be important to find a measurement for the harmfulness of absorbing sets in order to identify those ones which have the most detrimental effect on the decoding performance. Such a measurement would be extremly useful in code design for avoiding the most harmful absorbing sets. Generally, the benefit of a thorough trapping set analysis is twofold. Firstly, it provides deterministic methods for the prediction of error-floors over various channels. Secondly, it facilitates the design of structured LDPC codes with beneficial trapping set distributions and thus with good decoding performances, in particular, with very low error-floors.

A recent trend in coding theory is the construction and design of non-binary LDPC codes (e.g. [138]) which can outperform their binary counterparts in the case of small to moderate code lengths or in the case of higher order modulation [139]. However, these codes are still unsuitable for pratical usage due to a very high decoding complexity. An efficient hardware implementation of non-binary LDPC codes is still an open issue and only a few publications exist on this topic [140]. A promising approach is the algebraic design of structured non-binary LDPC codes on the basis of combinatorial tools such as finite fields and finite geometries (cf. [120]). This structure can potentially be exploited in order to leverage the implementation of such codes. However, the generalization of

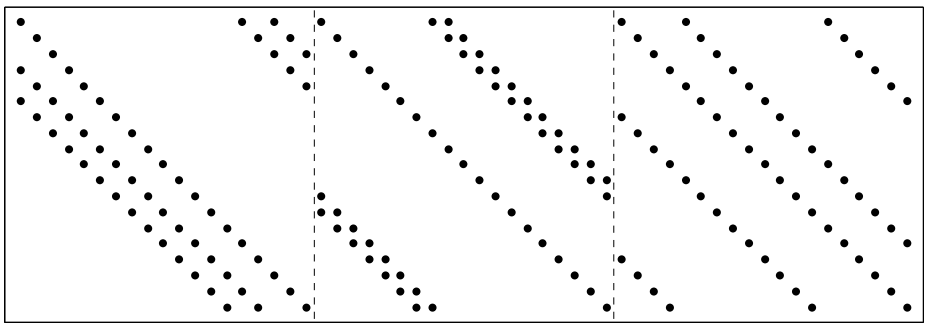
constructions from the binary to the non-binary case is a non-trivial problem. An interesting approach would be to transfer the techniques for the design of binary LDPC codes based on combinatorial designs to the non-binary case.

As shown in our paper [P4], the methods from the classical domain of structured high-rate LDPC code can also be exploited for the construction of quantum LDPC codes assisted by reliable qubits. The arising quantum LDPC codes inherit the characteristics of their classical counterparts, in particular, the low-complexity encoding along with the high decoding performances already at short to moderate code lengths. The results of our paper may encourage future studies to take advantage of classical code design and to find novel error-correction schemes by exploiting the phenomena unique to the world of quantum information [P4].

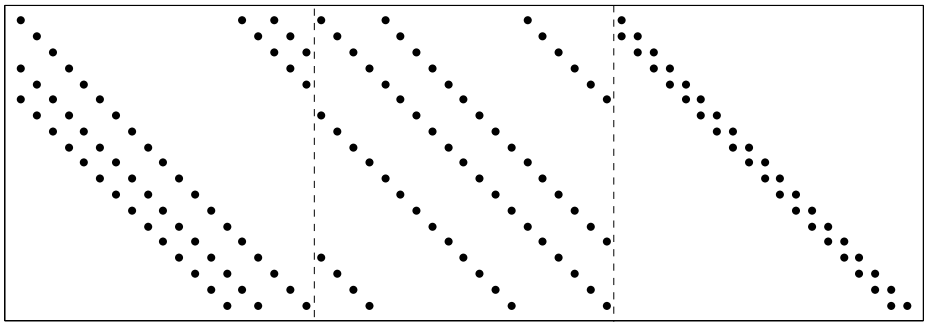
Appendices

A

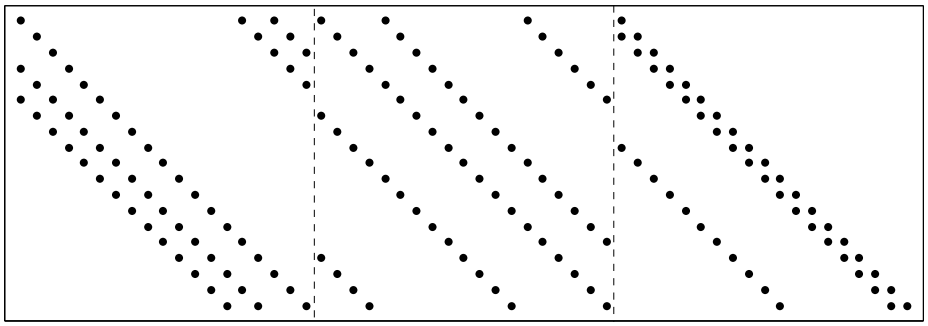
Examples of Structured Parity-Check Matrices



(a) CBIBD LDPC code as proposed in [35, 27]

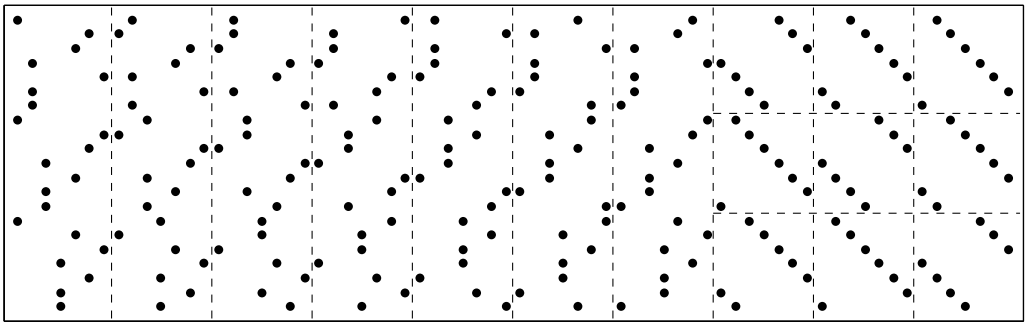


(b) CBIBD sRA code as proposed in Section 3.3.2

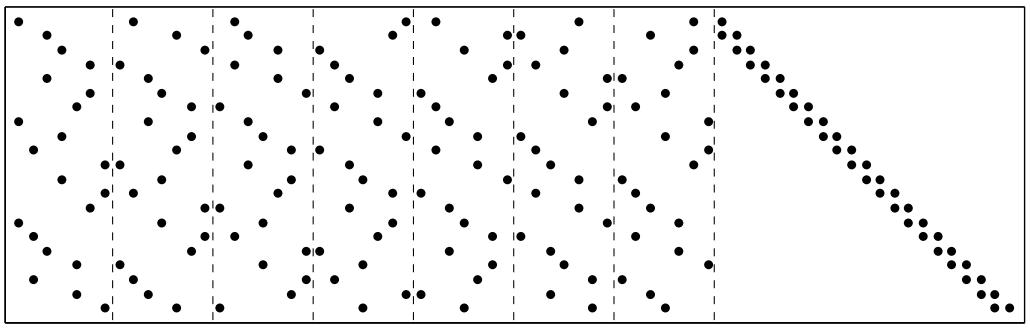


(c) CBIBD w3RA code as proposed in Section 3.3.2

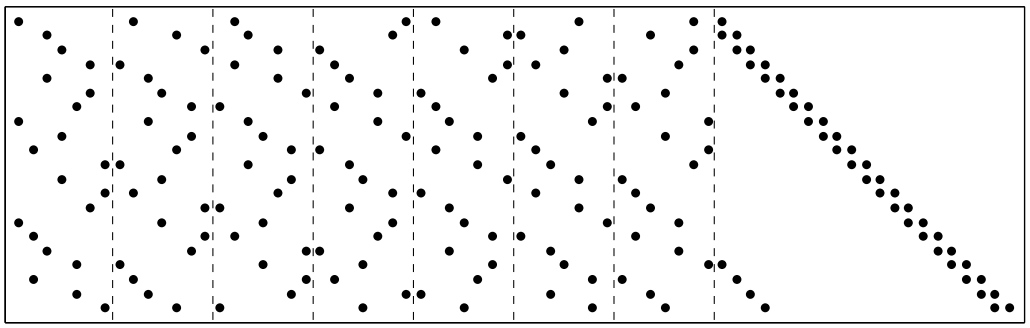
Abbildung A.1. – Parity-check matrix of a CBIBD LDPC, sRA and w3RA code of column weight $k = 3$, length $N = 57$ and rate $R = 0.67$ based on a CBIBD$(19, 3, 1)$ from Netto's first construction [92].



(a) RBIBD$(21, 3, 1)$ LDPC code as proposed in [33]. The underlying RBIBD is based on the construction of Ray-Chaudhuri and Wilson [94].

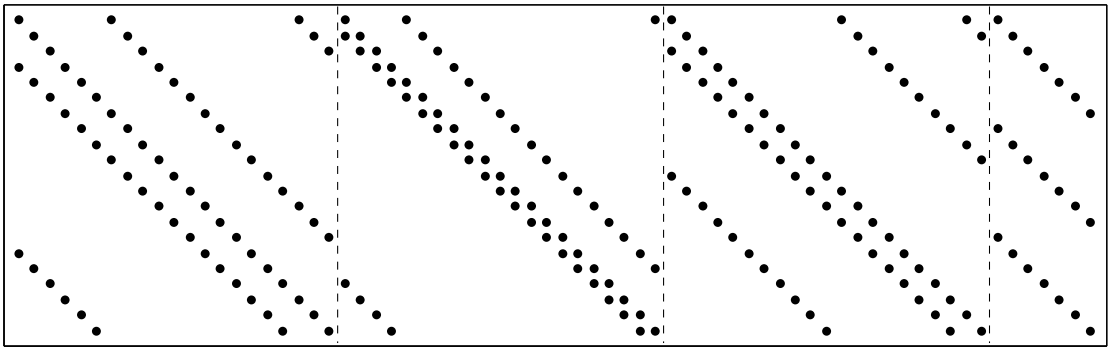


(b) RBIBD$(21, 3, 1)$ sRA code as presented in [10].

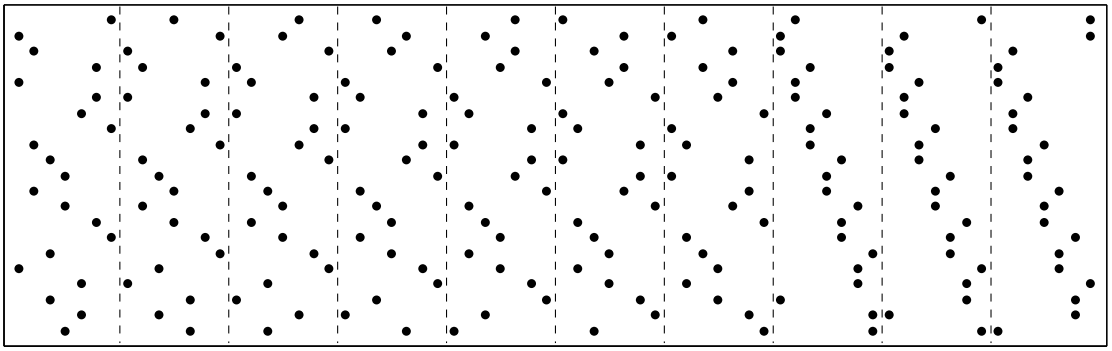


(c) RBIBD$(21, 3, 1)$ w3RA code as proposed in Section 3.3.3.

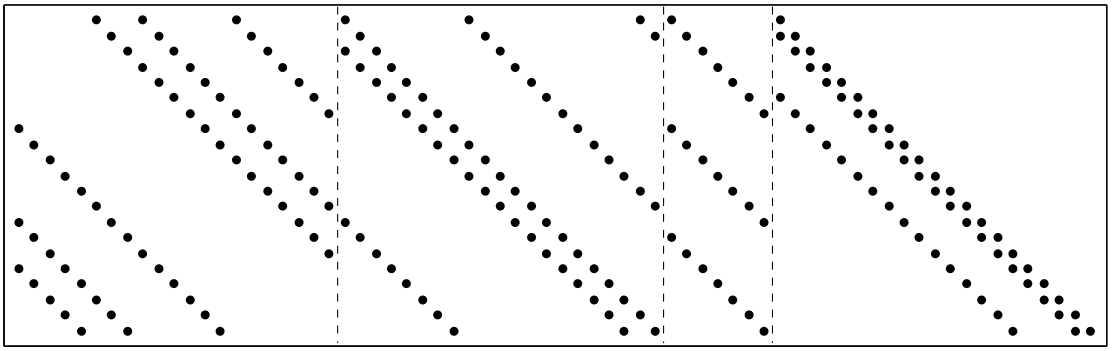
Abbildung A.2. – Parity-check matrix of an RBIBD LDPC, sRA and w3RA code of column weight $k = 3$, length $N = 70$ and rate $R = 0.7$ based on an RBIBD$(21, 3, 1)$ from the construction of Ray-Chaudhuri and Wilson.



(a) CRCBIBD LDPC code in cyclic representation



(b) CRCBIBD LDPC code in resolved representation



(c) CRCBIBD w3RA code in cyclic representation as proposed in Section 3.3.4.

Abbildung A.3. – Parity-check matrix of a CRCBIBD LDPC, sRA and w3RA code of column weight $k = 3$, length $N = 70$ and rate $R = 0.7$ based on a CRCBIBD$(21, 3, 1)$.

B

Classification Process

For the classification of SSCs and ASCs that may occur in TD LDPC codes, we have written a program that outputs an exhaustive list of non-isomorphic set systems which satisfy the combinatorial constraints (A)-(D) of Def. 13 in the case of SSCs and the constraints (A)-(D) of Def. 16 in the case of ASCs up to a given size of t blocks. By starting with an empty set system, we successively extend it by a further block (in all possible ways) in compliance with some combinatorial rules that are necessary to build up an SSC or ASC. When an extension fulfills all constraints of Def. 13 or Def. 16, we add it to the output list of SSCs or ASCs, respectively. We continue until a maximum number of t blocks has been reached.

$\mathcal{O} =$ **SSC/ASC_Classification**(k, t)

Input

* k: constant block size
* t: maximum number of blocks

Output

* $\mathcal{O}$: The output list $\mathcal{O}$ contains all non-isomorphic SSCs or ASCs of block size k and at most t blocks that may occur in any TD LDPC code of column weight k. The presence or absence of these entities in a TD LDPC code finally depends on the specific structure of the code.

Notations and Invariants

* The current set system is denoted by $\mathcal{S}$ with point set $\mathcal{P}$ and block set $\mathcal{B}$.
* A set system $(\mathcal{P}', \mathcal{B}')$ is called an *extension* of $\mathcal{S}$ if $\mathcal{P} \subseteq \mathcal{P}'$ and $\mathcal{B} \subseteq \mathcal{B}'$.
* Let $\mathcal{E}_\varpi$, $1 \leq \varpi \leq t$, be t global lists of non-isomorphic extensions of size ϖ that have already been processed. Note that the set systems collected in $\mathcal{E}_\varpi$ are not necessarily SSCs or ASCs.

Algorithm

(1) *Initialization*: We start with an empty set system $\mathcal{S}$. Define $\mathcal{E}_\varpi = \varnothing$ for $1 \leq \varpi \leq t$ and $\mathcal{O} = \varnothing$.

(2) *Find extensions of* $\mathcal{S}$: We extend $\mathcal{S}$ by a further block of size k in all possible ways with the following restrictions:[1]

(2a) The constraints (A)-(C) of Def. 13 (or Def. 16) must be valid.

(2b) $\mathcal{S}$ must be connected, i.e, for any two distinct subsets of blocks (called components) there must be at least one point that is contained in both components.

Let $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\mu$ be all possible non-isomorphic extensions of $\mathcal{S}$.

(3) *For each extension* $\mathcal{S}_i$ *with number of blocks* ϖ *do*:

(3a) If $\mathcal{S}_i$ is isomorphic to any set system of $\mathcal{E}_\varpi$, discard $\mathcal{S}_i$ and continue with the next extension, else add $\mathcal{S}_i$ to $\mathcal{E}_\varpi$.

(3b) Add $\mathcal{S}_i$ to the output $\mathcal{O}$, if $\mathcal{S}_i$ satisfies constraint (D) of Def. 13 in the case of SSCs or constraint (D) of Def. 16 in the case of ASCs.

(3c) If $\varpi = t$, we stop processing this extension, else we apply steps (2)-(3) recursively to $\mathcal{S} := \mathcal{S}_i$.[2]

---

[1] Note that any violation of (2a) is irreparable by extending the set system such that the concerned extensions can be discarded at this early stage. By contrast, the constraint (2b) is reparable by adding some new blocks properly such that the isolated components get connected. Nevertheless, we may discard unconnected set systems here since all connected extensions will be found by extending the blocks in different order. All constraints therefore reduce the processing complexity.

[2] Note that the violation of constraint (D) is in both cases reparable by adding some new blocks properly

such that the extensions must be further processed even if they violate (D).

C

Stopping Set Classification

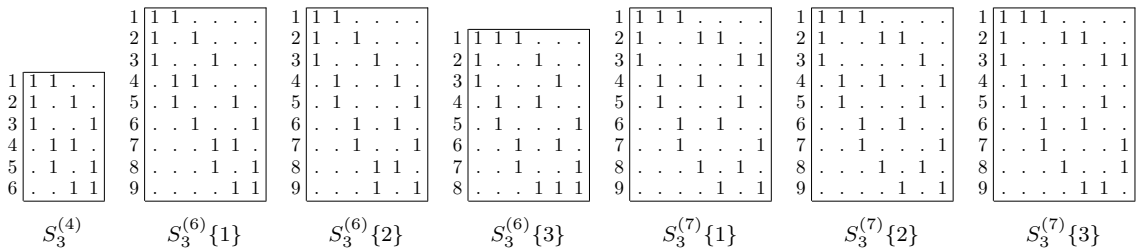


Abbildung C.1. – Matrix representation of all stopping set candidates $S_3^{(\ell)}\{i\}$ of size $\ell \leq 7$ and column weight three (enumerated by $\{i\}$) that may occur in a TD LDPC code of column weight three. The dots represent zero-entries.

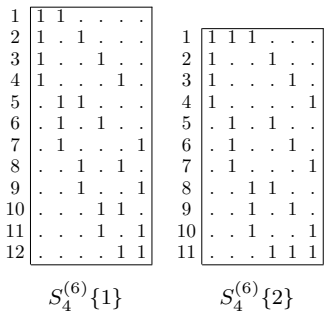


Abbildung C.2. – Complete list of all stopping set candidates of size $\ell \leq 7$ and column weight four (in matrix representation) that may occur in a TD LDPC code of column weight four. The dots represent zero-entries. Note that there are no SSCs of size $\ell \leq 5$ and of size seven.

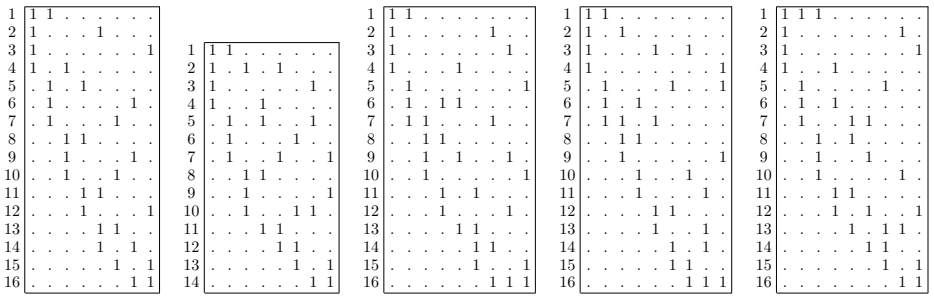


Abbildung C.3. – Matrix representation of stopping set candidates of size $\ell \in \{8, 9\}$ and column weight four that occur in at least one $\mathscr{L}_q^2$-TD LDPC code with $k = 4$ and $q \geq 5$. The dots represent zero-entries. The SSCs have been found by an extensive computer search.

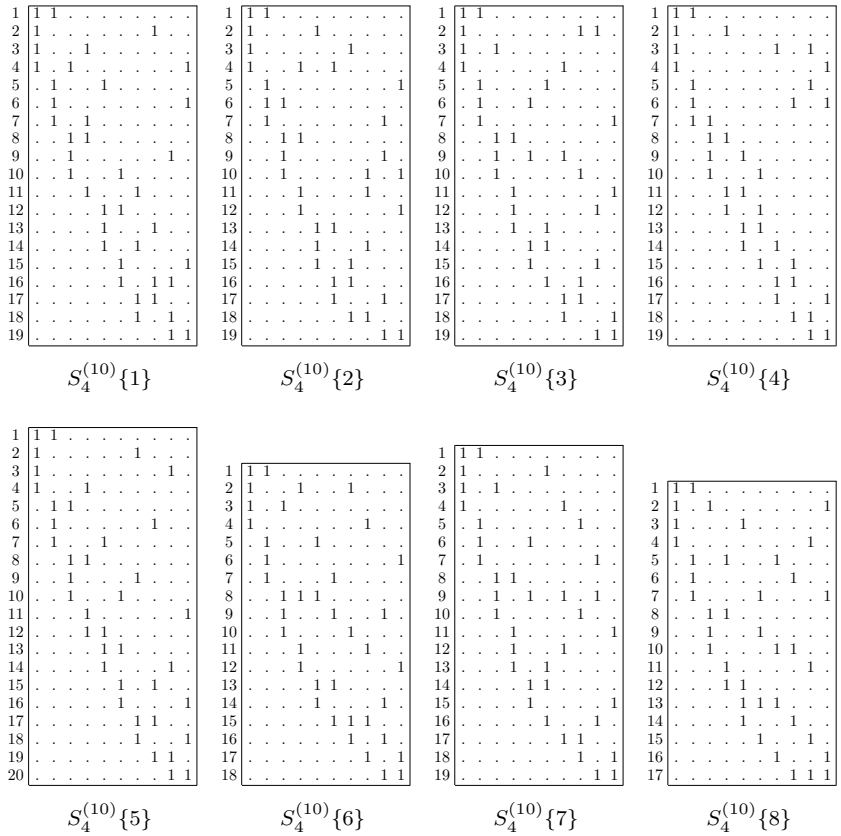


Abbildung C.4. – Matrix representation of harmful stopping set candidates of size ten and column weight four that occur in at least one $\mathscr{L}_q^m$-TD LDPC code with $k = 4$ and $q \geq 13$. The dots represent zero-entries. The SSCs have been found by an extensive computer search.

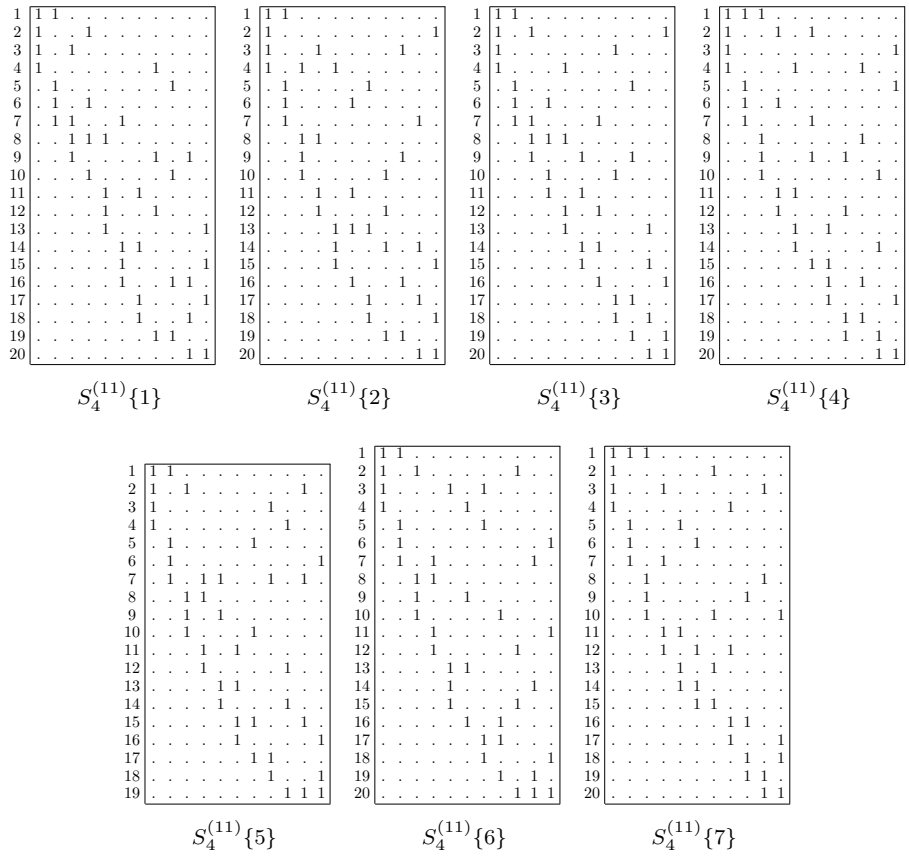


Abbildung C.5. – Matrix representation of stopping set candidates of size eleven and column weight four that occur in at least one $\mathscr{L}_q^2$-TD LDPC codes with $k = 4$ and $q \geq 13$. The dots represent zero-entries. The SSCs have been found by an extensive computer search.

D

Absorbing Set Classification

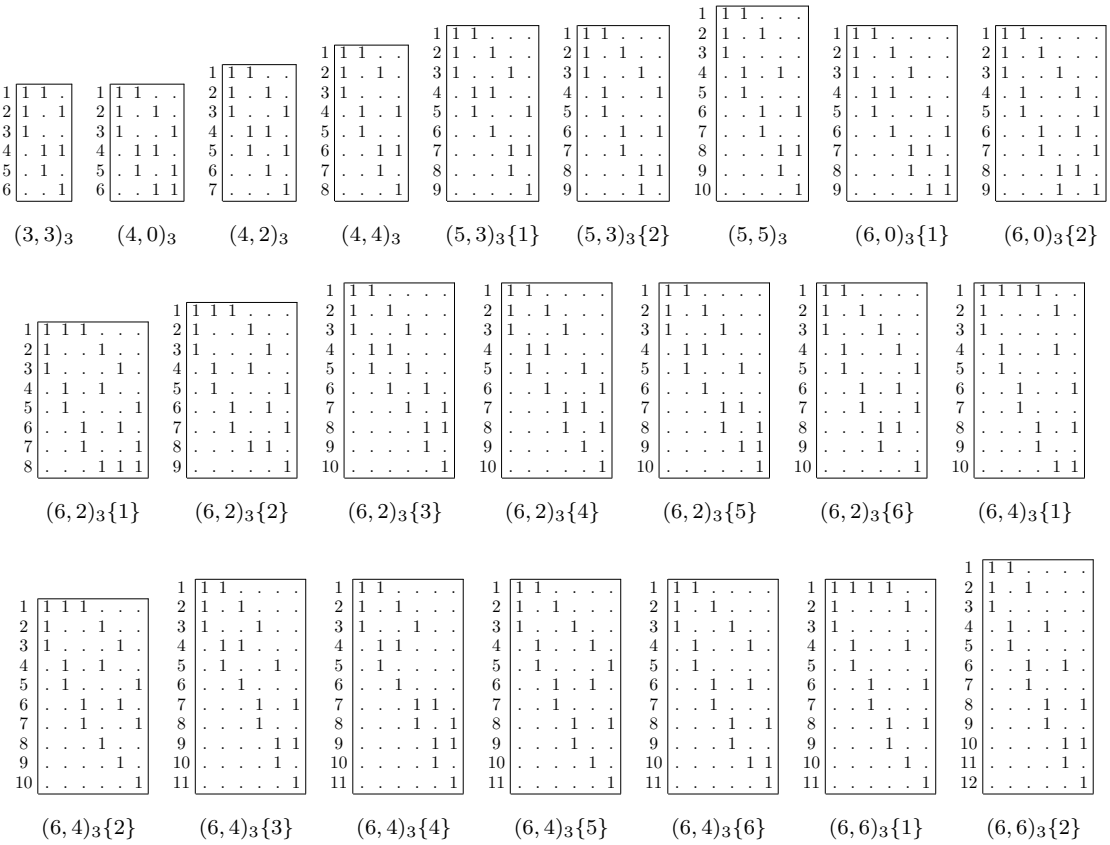


Abbildung D.1. – Complete list of $(a, b)_3$ absorbing set candidates of size $a \leq 6$, syndrome b and column weight three. All depicted candidates are 3-colorable. If there are multiple candidates of the same size $(a, b)_3$, we use a postfix $\{i\}$ to determine an order. The dots represent zero-entries.

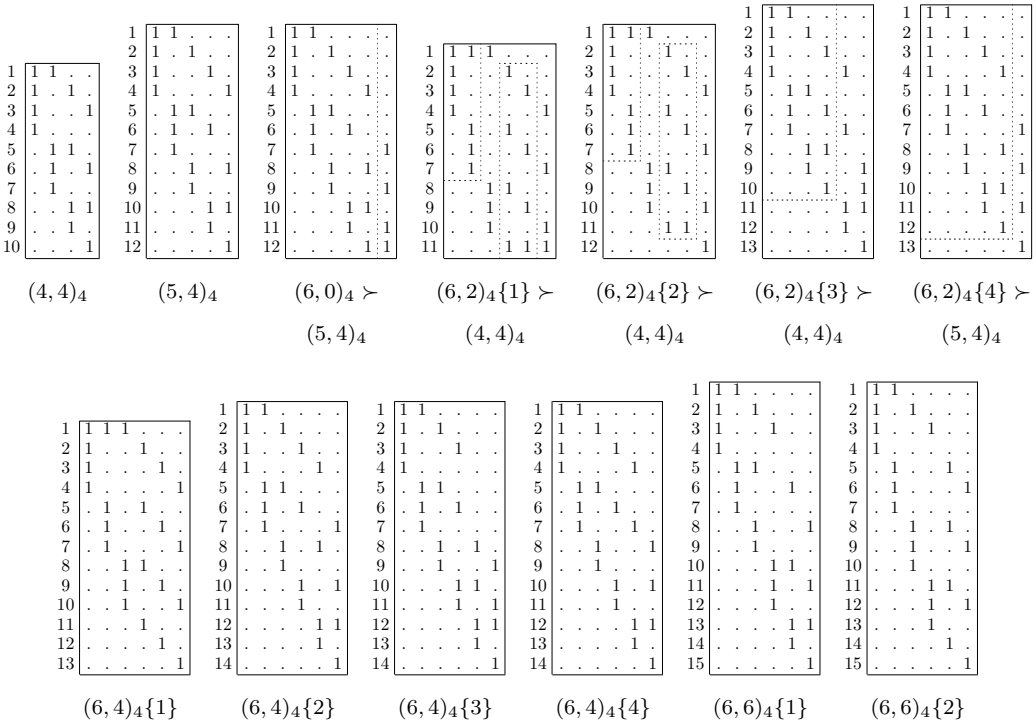


Abbildung D.2. – Complete list of $(a, b)_4$ absorbing set candidates of size $a \leq 6$, syndrome b and column weight four that may occur in a TD LDPC code of column weight four. All depicted candidates are 4-colorable. If there are multiple candidates of the same size $(a, b)_4$, we use a postfix $\{i\}$ to determine an order. The symbol "$\succ$" means that the left-hand candidate is an extension of the right-hand candidate. The dots represent zero-entries.

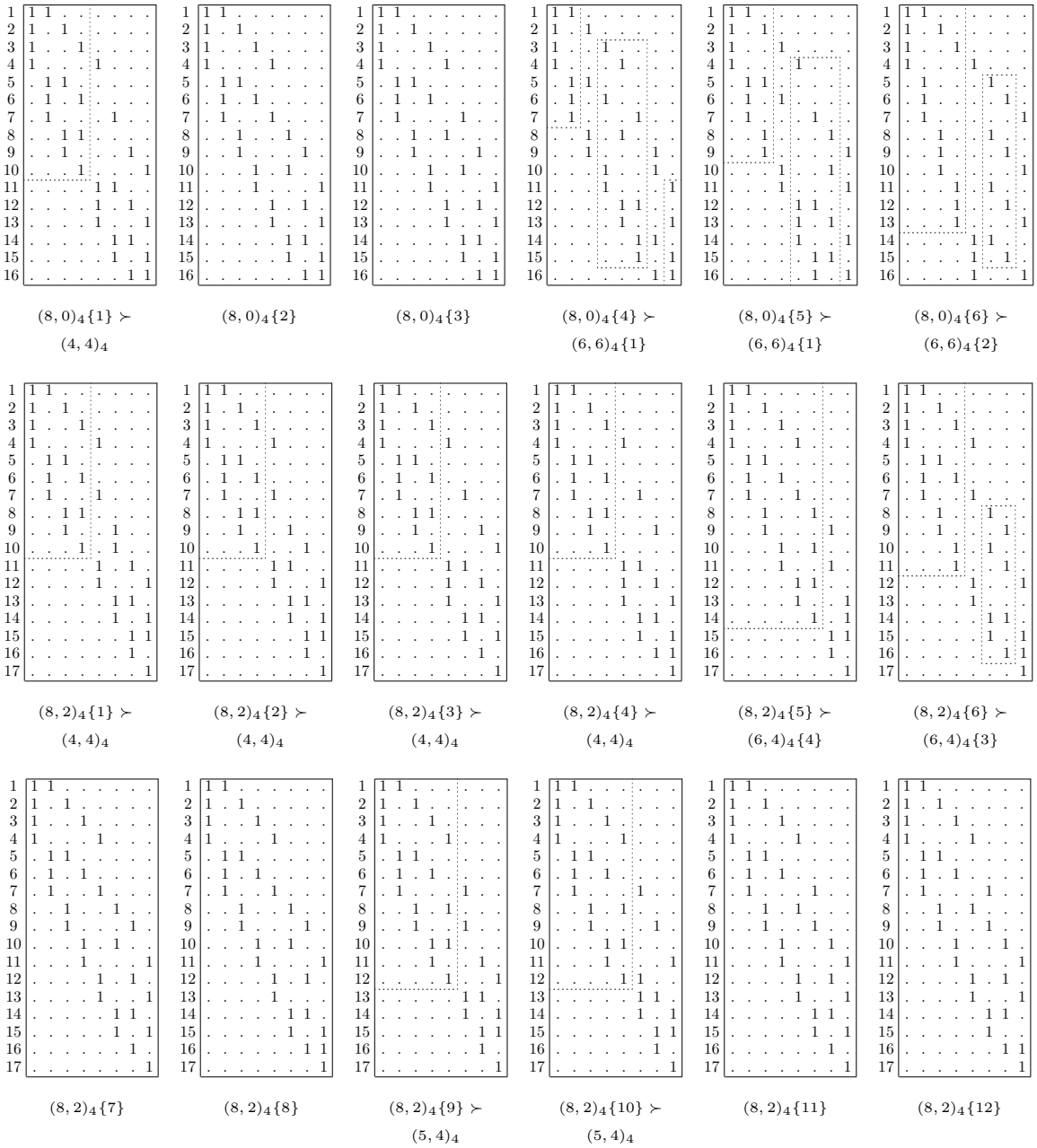


Abbildung D.3. – (Figure continues on the next page)

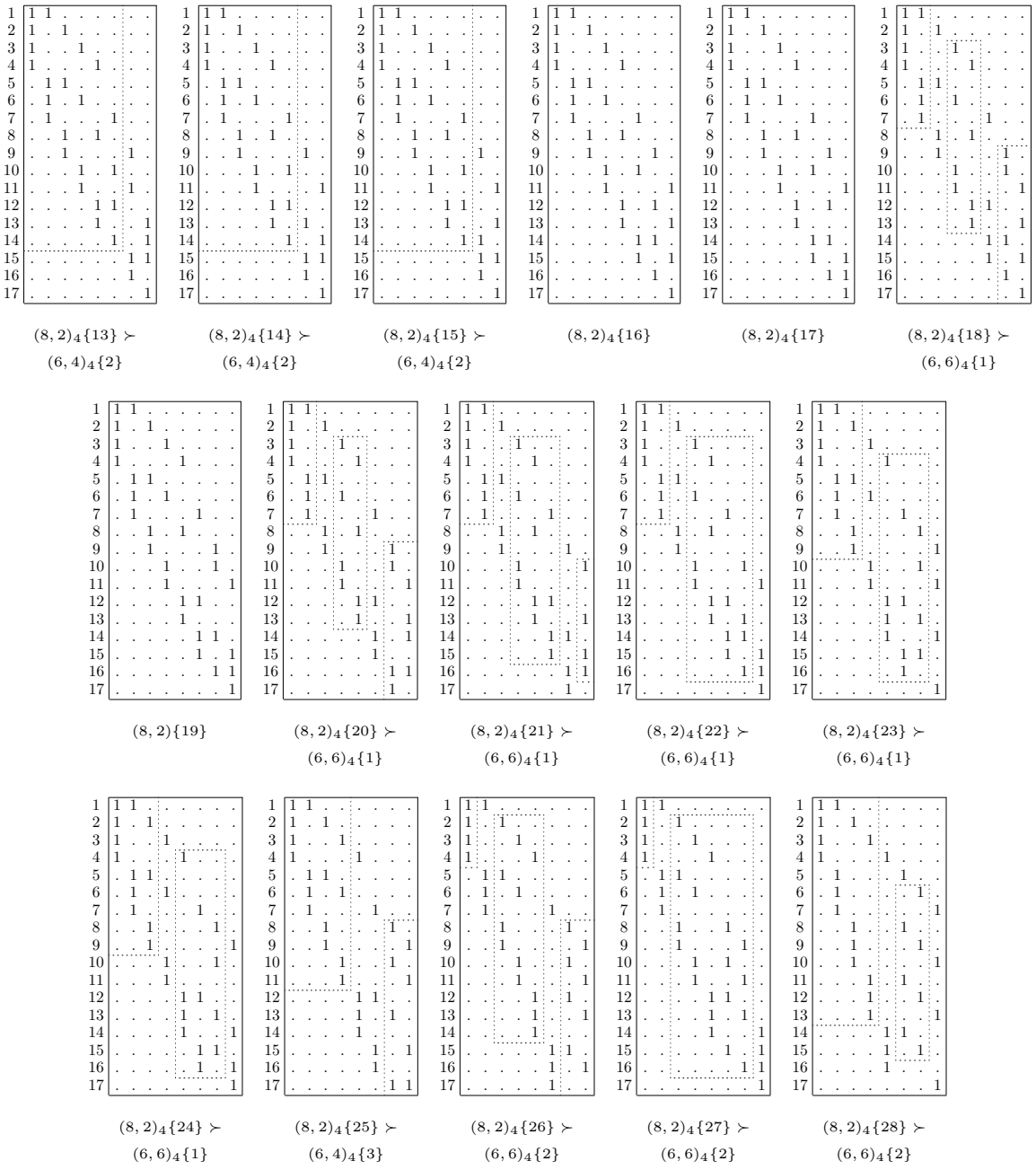


Abbildung D.3. – (Figure starts at the previous page). Complete list of all $(a, b)_4$ absorbing set candidates of size $a = 8$, syndrome $b \leq 2$ and column weight four that may occur in a TD LDPC code of column weight four. All depicted candidates are 4-colorable. If there are multiple candidates of the same size $(a, b)_4$, we use a postfix $\{i\}$ to determine an order. The symbol "$\succ$" means that the left-hand candidate is an extension of the right-hand candidate. The dots represent zero-entries.

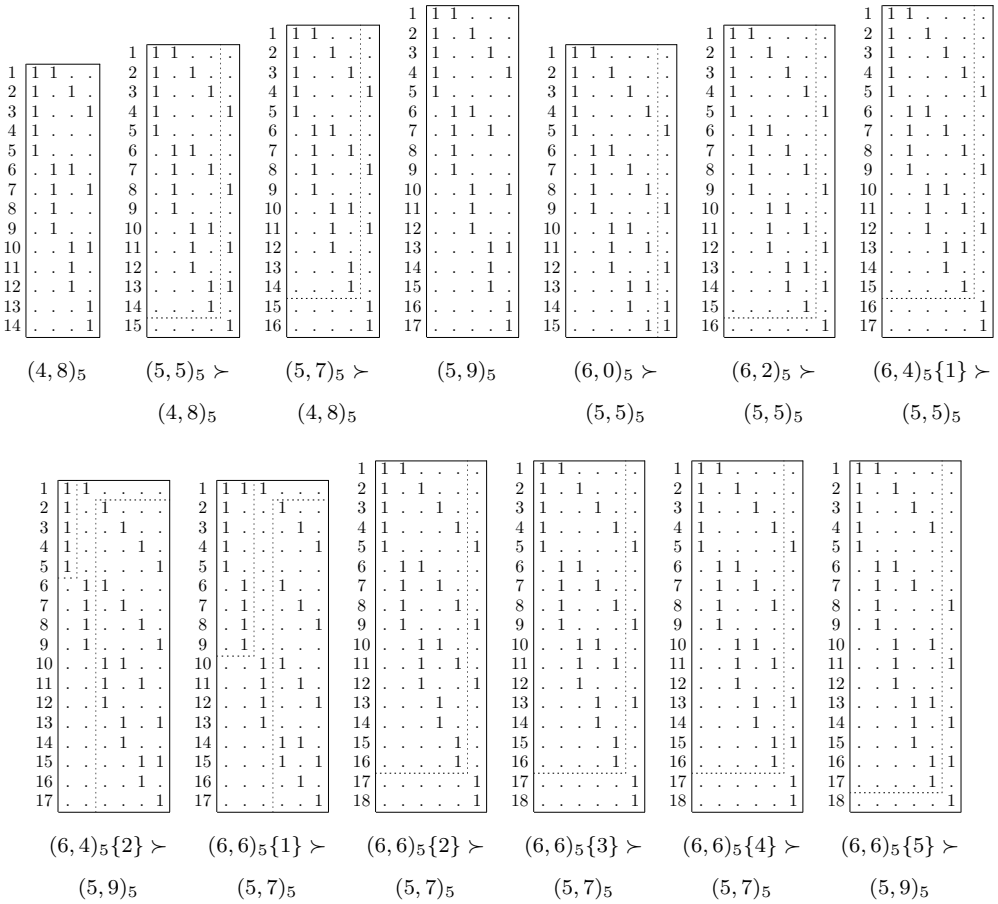


Abbildung D.4. – Complete list of all $(a,b)_5$ absorbing set candidates of size $a \leq 5$ and of all candidates with $a = 6$ and $b \leq 6$ that may occur in a TD LDPC code of column weight $k = 5$. All depicted candidates are 5-colorable. If there are multiple candidates of the same size $(a,b)_5$, we use a postfix $\{i\}$ to determine an order. The symbol "$\succ$" means that the left-hand candidate is an extension of the right-hand candidate. The dots represent zero-entries.

E

Elimination Process

Let $\mathcal{C}$ be an $\mathscr{L}_q^m$–TD LDPC code of block size $k := m + 2$ based on m MOLS of order q and scale factors $\alpha_1, \ldots, \alpha_m$. Furthermore, let $\mathcal{G} = \{G_1, \ldots, G_k\}$ be the groups of the underlying transversal design $\mathscr{L}_q^m$–TD such that G_1 and G_2 correspond to the common row and column set of the MOLS, respectively, and G_{i+2} corresponds to the symbol set of the i-th Latin square with scale factor α_i for $1 \leq i \leq m$.

Now, let $\mathcal{S}$ be an SSC (or ASC) with point set $\mathcal{P}$ and block set $\mathcal{B}$ and let φ be any k-coloring of $\mathcal{S}$. Clearly, there exist $(k!)$ possibilities to associate the k colors with the k groups of the transversal design. Let π be any bijective mapping, called *color-to-group mapping*, that maps the i-th color to the group G_i. Then, the following theorem holds.

Theorem 40 *There exists a stopping set (or absorbing set) of type $\mathcal{S}_\varphi$ with color-to-group mapping π in the factor graph of $\mathcal{C}$ if and only if there is an assignment of values from $\mathbb{F}_q$ to the variables $p_1, \ldots, p_\upsilon$ with $\upsilon := |\mathcal{P}|$ with the following properties.*

(1) The x-th point of $\mathcal{P}$ is associated with variable p_x.

(2) All elements of $\big\{\big(p_x, \varphi(x)\big) : x \in \mathcal{P}\big\}$ are unique.

(3) For every block $\{x_1, \ldots, x_k\} \in \mathcal{B}$ with $\pi(\varphi(x_1)) = G_1, \ldots, \pi(\varphi(x_k)) = G_k$, the linear equation $\alpha_i p_{x_1} + p_{x_2} - p_{x_{i+2}} = 0$ is satisfied over $\mathbb{F}_q$ for all $i = 1, \ldots, m$.

The linear equations obtained by Thm. 40 lead to a homogeneous linear system over $\mathbb{F}_q$ with unknown variables $p_1, \ldots, p_\upsilon$ and coefficients depending on the scale factors α_i.

Let $\boldsymbol{E}$ be the coefficient matrix and $\boldsymbol{p} = (p_1, \ldots, p_v)^T$ be the column vector of the unknown variables. Every solution of $\boldsymbol{Ep} = \boldsymbol{0}$ corresponds to a stopping (or absorbing) set of type $\mathcal{S}_\varphi$ if the pairs $\big\{\big(p_x, \varphi(x)\big) : x \in \mathcal{P}\big\}$ are unique. Conversely, if such a solution does not exist, there can not be any stopping (or absorbing) set of this type. Hence, the existence of any stopping (or absorbing) set of type $\mathcal{S}_\varphi$ with color-to-group mapping π in the factor graph of $\mathcal{C}$ is equivalent to the existence of certain solutions of the linear equation system $\boldsymbol{Ep} = \boldsymbol{0}$ obtained by Thm. 40. We solve this system symbolically by a modified Gaussian elimination algorithm in dependence of the scale factors α_i and the Galois field $\mathbb{F}_q$.

$e(\mathcal{S}_\varphi, \mathcal{C}) =$ **FindEliminationCondition**$(\mathcal{S}, \varphi, \mathcal{C})$

Input

* $\mathcal{S}$: the set system representing an SSC (or ASC) with constant block size k
* φ: any k-coloring of the set system $\mathcal{S}$
* $\mathcal{C}$: an $\mathscr{L}_q^{k-2}$–TD LDPC code based on MOLS of order q and scale factors $\alpha_1, \ldots, \alpha_m$

Output

* $e(\mathcal{S}_\varphi, \mathcal{C})$: the output is an elimination condition depending on the scale factors $\alpha_1, \ldots, \alpha_m$ of $\mathcal{C}$ such that all stopping (or absorbing) sets of type $\mathcal{S}_\varphi$ can be eliminated in $\mathcal{C}$ if and only if $e(\mathcal{S}_\varphi, \mathcal{C}) = 1$ over $\mathbb{F}_q$.

Notations and Invariants

* Let $\pi_1, \ldots, \pi_{k!}$ be all possible color-to-group mappings and let π be the current one, initialized by $\pi := \pi_1$. Furthermore, let $e_\pi(\mathcal{S}_\varphi, \mathcal{C})$ be the elimination condition for stopping (or absorbing) sets of type $\mathcal{S}_\varphi$ in $\mathcal{C}$ with color-to-group mapping π.
* Let μ be the identifier of the current process. Initially, we have $\mu = 0$.
* By a polynomial we mean an expression depending on the variables $\alpha_1, \ldots, \alpha_m$. We say that a polynomial is definitely zero, if the polynomial becomes zero after

evaluating the polynomial over $\mathbb{F}_q$ for every possible choice of the scale factors α_i and $\mathbb{F}_q$. By contrast, we say that the polynomial is definitely non-zero if the result becomes non-zero for every possible choice. For all other polynomials it depends on the values of α_i and q to decide whether the polynomial becomes zero or non-zero.

* The set $\Lambda_\mu^{(0)}$ contains all polynomials that are definitely zero in process μ. For $\mu = 0$, we initialize $\Lambda_0^{(0)} = \{0\}$.
* The set $\Lambda_\mu^{(+)}$ contains all polynomials that are definitely non-zero in process μ. Initially, we have[1]
  * $1, -1 \in \Lambda_0^{(+)}$,
  * $\alpha_i \in \Lambda_0^{(+)}$ for $1 \leq i \leq m$,
  * $(\alpha_i - \alpha_j) \in \Lambda_0^{(+)}$ for $1 \leq i, j \leq m$ and $i \neq j$,
  * $u \in \Lambda_0^{(+)}$ if u is not a prime power, and
  * $\lambda_1 \lambda_2 \in \Lambda_0^{(+)}$ if $\lambda_1, \lambda_2 \in \Lambda_0^{(+)}$.
* The set $\Lambda_\mu^{(*)}$ contains all polynomials that may be zero or non-zero depending on the choice of the scale factors α_i and $\mathbb{F}_q$. Consequently, it consists of all polynomials that are not in $\Lambda_\mu^{(0)} \cup \Lambda_\mu^{(+)}$.
* Let $\mathscr{T}$ be a rooted tree that consists of *case nodes* of the form $(\lambda, v)^{\mathrm{C}}$ and *elimination nodes* of the form $(\lambda, v)^{\mathrm{E}}$, where $\lambda \in \Lambda_\mu^{(*)}$, $v \in \{0, 1\}$ and where 'C' and 'E' stands for *case* and *elimination*, respectively. The tree shall be a global entity independent of any process. We say that a node is *satisfied* if $\varrho(\lambda, v) = 1$, where

$$\varrho(\lambda, v) = \begin{cases} 1, & \text{if } (\lambda \neq 0) \text{ and } (v = 1) \\ & \text{or } (\lambda = 0) \text{ and } (v = 0), \\ 0, & \text{otherwise.} \end{cases}$$

Note that it depends on the choice of the scale factors α_i and the choice of $\mathbb{F}_q$ if a

[1] Note that the scale factors $\alpha_i \in \mathbb{F}_q^*$ of any MOLS are non-zero and unique by definition such that α_i and any difference $\alpha_i - \alpha_j$ with $i \neq j$ are definitely non-zero. Furthermore, an entry that is equal to a prime power p^i with $i \geq 1$ can be zero if the underlying Galois field has characteristic p. Conversely, an entry that is not a prime power must be definitely non-zero. As an example, the term $6\alpha_1^2\alpha_2^3(\alpha_1 - \alpha_2)^2$ is definitely non-zero.

node is satisfied. Initialize the ouput tree $\mathscr{T}$ with an empty root node and assume that this root node is trivially satisfied.

* Let V_μ be the latest node of the current process μ, where V_0 is the root node of $\mathscr{T}$.

Algorithm

(1) Build up an equation system $\boldsymbol{Ep} = \boldsymbol{0}$ with the equations obtained by Thm. 40 for the set system $\mathcal{S}$ with k-coloring φ and color-to-group mapping π. Let $\boldsymbol{E}_\eta$ be the lower right submatrix of the coefficient matrix $\boldsymbol{E}$ including the η-th row and column. We start with $\eta = 1$.

(2) Find a column of $\boldsymbol{E}_\eta$ with all entries from $\Lambda_\mu^{(0)} \cup \Lambda_\mu^{(+)}$ and with at least one entry from $\Lambda_\mu^{(+)}$. If such a column does not exist, continue with step (3). Otherwise swap the rows and columns[2] of $\boldsymbol{E}$ in such a way that there is an entry of $\Lambda_\mu^{(+)}$ in the left top corner of $\boldsymbol{E}_\eta$. Then, apply row eliminations to $\boldsymbol{E}$ such that all other entries of the first column of $\boldsymbol{E}_\eta$ become zero (except the first) and continue with step (4).

(3) Find the column of $\boldsymbol{E}_\eta$ with the smallest positive number of entries in $\Lambda_\mu^{(*)}$. If such a column does not exist, i.e., if all entries are from $\Lambda_\mu^{(0)}$, continue with step (6). Otherwise, choose an entry $\lambda \in \Lambda_\mu^{(*)}$ of this column and do a case differentation. For this, split the current process into two subprocesses with identifier μ_1 and μ_2. For the first subprocess μ_1,

   * assume that $\lambda = 0$,
   * append the case node $(\lambda, 0)^{\mathrm{C}}$ to V_μ,
   * mark this node as the latest node V_{μ_1} of process μ_1,
   * define $\Lambda_{\mu_1}^{(0)} := \Lambda_\mu^{(0)} \cup \{\lambda\}$, and
   * continue recursively with step (5) and the process id $\mu := \mu_1$.

   For the second subprocess,

   * assume that $\lambda \neq 0$,
   * append the case node $(\lambda, 1)^{\mathrm{C}}$ to V_μ,
   * mark this node as the latest node V_{μ_2} of the process μ_2,

[2] If we swap rows of $\boldsymbol{E}$, we must also swap the corresponding entries in $\boldsymbol{p}$.

- define $\Lambda_{\mu_2}^{(+)} := \Lambda_{\mu}^{(+)} \cup \{\lambda\}$, and
- continue recursively with step (5) and the process id $\mu := \mu_2$.

(4) After every step search for linear equations of the form $\lambda(p_{x_1} - p_{x_2}) = 0$ with $\varphi(x_1) = \varphi(x_2)$ and $\lambda \in \Lambda_{\mu}^{(*)}$. By choosing the scale factors α_i and $\mathbb{F}_q$ in such a way that $\lambda \neq 0$ over $\mathbb{F}_q$, it follows that $(p_{x_1}, \varphi(x_1)) = (p_{x_2}, \varphi(x_2))$ which violates the second condition of Thm. 40 for all possible solutions. Hence, all stopping (or absorbing) sets of type $\mathcal{S}_\varphi$ with color-to-group mapping π can be eliminated by a proper choice of the scale factors α_i and $\mathbb{F}_q$. Consequently, append the elimination node $(\lambda, 1)^{\mathrm{E}}$ to the path node V_μ and, for further processing,

- assume that $\lambda = 0$
- remove the row of $\boldsymbol{E}$ that corresponds to the detected equation,
- set $\Lambda_{\mu}^{(0)} := \Lambda_{\mu}^{(0)} \cup \{\lambda\}$, and
- continue with step (5).

(5) Repeat step (2) with $\eta := \eta+1$ until the matrix is in row echelon form. If the matrix is in row echelon form, we solve the system symbolically by back substitution such that all unknown variables depend on the scale factors α_i and on some free variables if the system is underdetermined. We obtain symbolic expressions for $p_1, \ldots, p_v$.

(6) Compute the symbolic differences $p_{x_1} - p_{x_2}$ for all $x_1, x_2 \in \mathcal{P}$ with $\varphi(x_1) = \varphi(x_2)$ and evaluate under which conditions these differences are zero. More precisely, search for differences of the form $p_{x_1} - p_{x_2} = \lambda(p_{x_3} - p_{x_4}) = 0$ with $\lambda \in \Lambda_{\mu}^{(*)}$ and $x_3, x_4 \in \mathcal{P} \setminus \{x_1, x_2\}$. If we choose α_i and $\mathbb{F}_q$ in such a way that $\lambda = 0$, the points $(p_{x_1}, \varphi(x_1))$ and $(p_{x_2}, \varphi(x_2))$ coincide for all possible solutions and thus cannot lead to a stopping (or absorbing) set of type $\mathcal{S}_\varphi$ with color-to-group mapping π. As a consequence, $\lambda = 0$ is an elimination condition such that we can append the node $(\lambda, 0)^{\mathrm{E}}$ to V_μ.

(7) Let $\mathfrak{p} = \big(\mathrm{root}, (\lambda_1, v_1)^{\mathrm{C}}, (\lambda_2, v_2)^{\mathrm{C}}, \ldots, (\lambda_{n-1}, v_{n-1})^{\mathrm{C}}, (\lambda_n, v_n)^{\mathrm{E}}\big)$ be a path of $\mathscr{T}$ that ends up in an elimination node. Define $\varrho(\mathfrak{p}) := \bigwedge_{i=1}^{n} \varrho(\lambda_i, v_i)$. Then, the stopping (or absorbing) sets of type $\mathcal{S}_\varphi$ with color-to-group mapping π can be eliminated if $\varrho(\mathfrak{p}) = 1$. Let $\mathfrak{p}_1, \ldots, \mathfrak{p}_\xi$ be all paths of $\mathscr{T}$ that ends up in an elimination node.

Define $e_\pi(\mathcal{S}_\varphi, \mathcal{C}) := \bigvee_{i=1}^{\xi} \varrho(\mathfrak{p}_i)$. Then, the stopping sets (or absorbing sets) of type $\mathcal{S}_\varphi$ with color-to-group mapping π can be eliminated if and only if $e_\pi(\mathcal{S}_\varphi, \mathcal{C}) = 1$. Next, simplify the elimination condition $e_\pi(\mathcal{S}_\varphi, \mathcal{C})$ by using the following rules:

- $\varrho(\varepsilon\lambda, v) = \varrho(\lambda, v)$,
- $\varrho(\lambda^i, v) = \varrho(\lambda, v)$,
- if $\varrho(\lambda, v) \Rightarrow \varrho(\lambda', v')$, then $\varrho(\lambda, v) \vee \varrho(\lambda', v') = \varrho(\lambda', v')$,
- if $\varrho(\lambda, v) \Rightarrow \varrho(\lambda', v')$, then $\varrho(\lambda, v) \wedge \varrho(\lambda', v') = \varrho(\lambda, v)$,
- $\varrho(\lambda, 0) \vee \varrho(\lambda', 0) = \varrho(\lambda\lambda', 0)$
- $\varrho(\lambda\lambda', 1) \vee \varrho(\lambda, 1) = \varrho(\lambda, 1)$
- $\varrho(\lambda, 1) \vee \varrho(\lambda', 0) = \varrho(\lambda, 1) \vee \varrho(\varepsilon\lambda + \varepsilon'\lambda', 0)$,
- $\varrho(\lambda, 1) \vee \varrho(\lambda', 1) = \varrho(\lambda, 1) \vee \varrho(\varepsilon\lambda + \varepsilon'\lambda', 1)$,
- $\varrho(\lambda, 0) \wedge \varrho(\lambda', 0) = \varrho(\lambda, 0) \wedge \varrho(\varepsilon\lambda + \varepsilon'\lambda', 0)$,
- $\varrho(\lambda, 0) \wedge \varrho(\lambda', 1) = \varrho(\lambda, 0) \wedge \varrho(\varepsilon\lambda + \varepsilon'\lambda', 1)$,

where $\lambda, \lambda' \in \Lambda_0^{(*)}$, $\varepsilon, \varepsilon' \in \Lambda_0^{(+)}$, and $v, v' \in \{0, 1\}$.

(8) Repeat the entire process for each remaining color-to-group mapping $\pi_2, \ldots, \pi_{k!}$. The output elimination condition for the stopping (or absorbing) sets of type $\mathcal{S}_\varphi$ is then given by $e(\mathcal{S}_\varphi, \mathcal{C}) = \bigvee_{i=1}^{k!} e_{\pi_i}(\mathcal{S}_\varphi, \mathcal{C})$.

F

Proofs

F.1. Proof of Proposition 25

We first need the following Lemma:

Lemma 41 *Let $PG(2,q) = (\mathcal{P}', \mathcal{L}')$ be any projective subplane of $PG(m,q) = (\mathcal{P}, \mathcal{L})$ such that $\mathcal{P}' \subset \mathcal{P}$ and $\mathcal{L}' \subset \mathcal{L}$. Then, for any line $L \in \mathcal{L}$ it holds either that $L \in \mathcal{L}'$ or $|L \cap \mathcal{P}'| \leq 1$.*

Proof. For $|L \cap \mathcal{P}'| \geq 2$, let p_1, p_2 be any two points of $L \cap \mathcal{P}'$. It holds that $L = \langle p_1, p_2 \rangle$ since L is a 2-dimensional subspace of $\mathbb{F}_q^{m+1}$ and thus, $\langle p_1, p_2 \rangle \subset \mathcal{P}' = \langle p_1, p_2, p_3 \rangle$ for any $p_3 \in \mathcal{P}' \setminus L$. Hence, L must be completely contained in $\mathcal{P}'$ (which is equivalent to $L \in \mathcal{L}'$) for $|L \cap \mathcal{P}'| \geq 2$. ■

Now, let $\mathcal{E}$ be the lines of any stopping set in $PG(m,q)$. We construct a projective subplane $PG(2,q) = (\mathcal{P}', \mathcal{L}')$ in such a way that $|\mathcal{E} \cap \mathcal{L}'| \geq 2$. For this, choose any two lines of $\mathcal{E}$ having an intersection point p. Then take p and any other point from each of the both lines, giving a total of three points. The subspace spanned by these points gives the point set $\mathcal{P}'$, and $\mathcal{L}'$ consists of all lines of $\mathcal{L}$ that are completely contained in $\mathcal{P}'$. Define $\mathcal{D} := \mathcal{E} \cap \mathcal{L}'$ and $d := |\mathcal{E} \cap \mathcal{L}'|$. Now, we distinguish between the following three cases:

(1) *Case* $d = |\mathcal{E}|$: Since $\mathcal{E} \subseteq \mathcal{L}'$, $\mathcal{E}$ is trivially a stopping set in the projective subplane formed by $\mathcal{L}'$.

(2) *Case* $d < |\mathcal{E}|$ *and* $2 \leq d \leq q$: First, we define that the *degree* of a point with respect to a given subset of lines is the number of these lines that contain the point. In particular note that for a stopping set $\mathcal{E}$, each point of $\mathcal{P}_\mathcal{E} := \bigcup \mathcal{E}$ has degree of at least two with respect to the lines of $\mathcal{E}$. Let $\mathcal{M}$ be the subset of points of $\mathcal{P}_\mathcal{D} := \bigcup \mathcal{D}$ having degree one with respect to the lines of $\mathcal{D}$ and define $\mu := |\mathcal{M}|$. For every point $p_i \in \mathcal{M}$, there must be at least one line in $\mathcal{E} \setminus \mathcal{D}$ that contains p_i, such that each point of $\mathcal{M}$ has degree of at least two with respect to $\mathcal{E}$. Since the lines of $\mathcal{E} \setminus \mathcal{D}$ are not in $\mathcal{L}'$, they contain at most one point of $\mathcal{M} \subseteq \mathcal{P}_\mathcal{D}$ (cf. Lemma 41) and thus $|\mathcal{E} \setminus \mathcal{D}| \geq \mu$. Hence, we have $|\mathcal{E}| = |\mathcal{D}| + |\mathcal{E} \setminus \mathcal{D}| \geq d + \mu$. By considering the simple argument that any two lines of $\mathcal{D}$ have at most one point in common, the minimum value for μ can be reached if any two lines of $\mathcal{D}$ share exactly one point. This case is possible since there are less than $q+1$ lines in $\mathcal{D}$. Hence, a lower bound for μ can be established by

$$\mu \geq d(q+1) - 2\sum_{j=1}^{d-1} j = d(q-d+2).$$

Subsequently, it follows that

$$|\mathcal{E}| \geq d + \mu \geq d(q-d+3) := f(d).$$

It can be shown that $f(d) \geq 2q+2$ for $2 \leq d \leq q$ with minimum $f(2) = 2q+2$. Therefore, we have $|\mathcal{E}| \geq 2q+2$.

(3) *Case* $d < |\mathcal{E}|$ *and* $d \geq q+1$: Choose any line $L \in \mathcal{E} \setminus \mathcal{D}$, which must exist, since $d < |\mathcal{E}|$. L intersects with $\mathcal{P}_\mathcal{D}$ in at most one point (Lemma 41) such that at least q points from L (with $|L| = q+1$) are in $\mathcal{P}_\mathcal{E} \setminus \mathcal{P}_\mathcal{D}$. Hence, there must be at least q further lines in $\mathcal{E} \setminus \mathcal{D}$ such that every point of L occurs in at least two lines of $\mathcal{E}$, giving $|\mathcal{E} \setminus \mathcal{D}| \geq q+1$. Consequently, we have $|\mathcal{E}| = |\mathcal{D}| + |\mathcal{E} \setminus \mathcal{D}| \geq d+q+1 \geq 2q+2$.

F.2. Proof of Proposition 34

First, we need the following straightforward Lemma:

Lemma 42 *Let $AG(2,q) = (\mathcal{P}', \mathcal{L}')$ be an affine subplane of $AG(m,q) = (\mathcal{P}, \mathcal{L})$ such that $\mathcal{P}' \subset \mathcal{P}$ and $\mathcal{L}' \subset \mathcal{L}$. For any line $L \in \mathcal{L}$ it holds either that $L \in \mathcal{L}'$ or $|L \cap \mathcal{P}'| \leq 1$.*

Proof. For $|L \cap \mathcal{P}'| \geq 2$, let p_1, p_2 be any two points of $L \cap \mathcal{P}'$. The line L is a coset of a 1-dimensional subspace of $\mathbb{F}_q^m$ and can be described by $L = \{\alpha(p_2 - p_1) + p_1 : \alpha \in \mathbb{F}_q\}$. Clearly, $p_1 \in L$ (with $\alpha = 0$) and $p_2 \in L$ (with $\alpha = 1$). The point set $\mathcal{P}'$ is a coset of a 2-dimensional subspace of $\mathbb{F}_q^m$ and can be described by $\mathcal{P}' = \{\alpha(p_2 - p_1) + \beta(p_3 - p_1) + p_1 : \alpha, \beta \in \mathbb{F}_q\}$ for any $p_3 \in \mathcal{P}' \setminus (L \cup \{0, \ldots, 0\})$. It can easily be seen that $L \subset \mathcal{P}'$ by setting $\beta = 0$. ■

Now, let $\mathcal{E}$ be the lines of any stopping set in $AG(m,q)$. We construct an affine subplane $PG(2,q) = (\mathcal{P}', \mathcal{L}')$ such that $|\mathcal{E} \cap \mathcal{L}'| \geq 2$. For this, choose any two lines of $\mathcal{E}$ having an intersection point p_1. Then take p_1 and one other point from each of the both lines, say p_2 and p_3. Then, the point set of the subplane can be constructed by $\mathcal{P}' = \{\alpha(p_2 - p_1) + \beta(p_3 - p_1) + p_1 : \alpha, \beta \in \mathbb{F}_q\}$ and $\mathcal{L}'$ consists of all lines of $\mathcal{L}$ whose points are all contained in $\mathcal{P}'$. Now, we distinguish between the following three cases:

(1) *Case $d = |\mathcal{E}|$*: Since $\mathcal{E} \subseteq \mathcal{L}'$, $\mathcal{E}$ is trivially a stopping set in the affine subplane formed by $\mathcal{L}'$.

(2) *Case $d < |\mathcal{E}|$ and $2 \leq d \leq q$*: Let $\mathcal{M}$ be the subset of points of $\mathcal{P}_\mathcal{D} := \bigcup \mathcal{D}$ that occur in exactly one line of $\mathcal{D}$, and define $\mu := |\mathcal{M}|$. By considering that any two lines of $\mathcal{D}$ can have at most one point in common, it can be established that

$$\mu \geq dq - 2\sum_{j=1}^{d-1} j = d(q - d + 1).$$

Since $\mathcal{E}$ is a stopping set, every point $p_i \in \mathcal{M}$ must occur in at least two lines of $\mathcal{E}$ and thus there must be at least one line in $\mathcal{E} \setminus \mathcal{D}$ that contains p_i. Since the

lines of $\mathcal{E} \setminus \mathcal{D}$ are not in $\mathcal{L}'$, each line contains at most one point of $\mathcal{M}$ according to Lemma 42. Hence, $|\mathcal{E} \setminus \mathcal{D}| \geq \mu$. It follows that $|\mathcal{E}| = |\mathcal{D}| + |\mathcal{E} \setminus \mathcal{D}| \geq d + \mu$ and subsequently that

$$|\mathcal{E}| \geq d + \mu \geq d(q - d + 2) := f(d).$$

It can be shown that $f(d) \geq 2q$ for $2 \leq d \leq q$ with minimum $f(2) = f(q) = 2q$. Hence, we have $|\mathcal{E}| \geq 2q$.

(3) *Case* $d < |\mathcal{E}|$ *and* $d \geq q + 1$: Choose any line $L \in \mathcal{E} \setminus \mathcal{D}$, which must exist, since $d < |\mathcal{E}|$. L intersects with $\mathcal{P}_\mathcal{D}$ in at most one point (Lemma 42) such that at least $q - 1$ points from L (with $|L| = q$) are in $\mathcal{P}_\mathcal{E} \setminus \mathcal{P}_\mathcal{D}$. Hence, there must be at least $q-1$ further lines in $\mathcal{E} \setminus \mathcal{D}$ such that every point of L occurs in at least two lines of $\mathcal{E}$, giving $|\mathcal{E} \setminus \mathcal{D}| \geq q$. Consequently, we have $|\mathcal{E}| = |\mathcal{D}| + |\mathcal{E} \setminus \mathcal{D}| \geq d + q \geq 2q + 1$.

Frequently Used Symbols

$\mathcal{C}$	linear block code
H	parity-check matrix
N	code length
M	number of parity-check equations
k	column weight
r	row weight
R	code rate
R_d	design rate
ρ_H	density of the parity-check matrix H
$d_{\min}(\mathcal{C})$	minimum distance of the code $\mathcal{C}$
$s_{\min}(H)$	stopping distance of a code's parity-check matrix H
H_1	first submatrix of an sRA code's parity-check matrix
H_2	second submatrix of an sRA code's parity-check matrix
a	constant row weights of H_1
q	constant column weights of H_1
Π	interleaver of an sRA code

Abbreviations

LDPC	low-density parity-check code
RA code	repeat-accumulate code
sRA code	systematic RA code
wqRA code	weight-q RA code
BIBD	balanced incomplete block design
CBIBD	cyclic BIBD
RBIBD	resolvable BIBD
CRCBIBD	cyclically resolvable cyclic BIBD
CDF	cyclic difference family
RDF	radical difference family
MOLS	mutually orthogonal Latin squares
TD	transversal design
BEC	binary erasure channel
BSC	binary symmetric channel
AWGN	additive white Gaussian noise
SNR	signal-to-noise ratio
BER	bit error rate
SPA	sum-product algorithm
SSC	stopping set candidate
ASC	absorbing set candidate
FG	finite geometry
PG	projective geometry
AG	affine geometry

Literaturverzeichnis

[1] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[2] A. Shokrollahi, "LDPC Codes: An Introduction," 2003. [Online]. Available: https://www.ics.uci.edu/~welling/teaching/ICS279/LPCD.pdf

[3] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[4] ——, "Low-Density Parity-Check Codes," Ph.D. dissertation, MIT Press, Cambridge, MA, 1963.

[5] T. Richardson and R. Urbanke, *Modern Coding Theory.* Cambridge University Press, 2008.

[6] G. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding: Turbo codes," in *Proc. 1993 Int. Conf. Comm.*, Geneva, Switzerland, 1993, pp. 1064–1070.

[7] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. on Communications*, vol. 44, no. 10, pp. 1261–1271, 1996.

[8] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding, 5th IMA Conference (Lecture Notes in Computer Science)*, C. Boyd, Ed., vol. 1025, 1995, pp. 110–111.

[9] D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for turbo-like codes," in *Proc. 1998 Allerton Conf. Commun., Control Comput.*, 1998, pp. 201–210.

[10] S. J. Johnson and S. R. Weller, "Combinatorial Interleavers for Systematic Regular Repeat-Accumulate Codes," *IEEE Trans. on Communications*, vol. 56, no. 8, pp. 1201–1206, 2008.

[11] ——, "Practical interleavers for repeat-accumulate codes," *IEEE Trans. on Communications*, vol. 57, no. 5, pp. 1225–1228, 2009.

[12] Y. Wang and J. S. Yedidia and S. C. Draper, "Construction of high-girth QC-LDPC codes," in *2008 5th Int. Symp. on Turbo Codes and Related Topics*, Sept. 2008, pp. 180–185.

[13] L. Dolecek, Z. Zhang, V. Anantharam, M. J. Wainwright and B. Nikolić, "Analysis of Absorbing Sets and Fully Absorbing Sets of Array-Based LDPC Codes," *IEEE Trans. on Information Theory*, vol. 56, no. 1, pp. 181–201, 2010.

[14] M. Eroz, F. Sun and F. Lee, "DVB-S2 low density parity check codes with near Shannon limit performance," *International Journal of Satellite Communications and Networking*, vol. 22, no. 3, pp. 269–279, 2004.

[15] S. J. Johnson, *Iterative Error Correction: Turbo, Low-Density Parity-Check and Repeat-Accumulate Codes*. Cambridge University Press, 2009.

[16] J. Kim and A. Ramamoorthy and S. W. McLaughlin, "The design of efficiently-encodable rate-compatible LDPC codes," *IEEE Trans. on Communications*, vol. 57, no. 2, pp. 365–375, 2009.

[17] E. R. Berlekamp, *Algebraic coding theory*. Laguna Hills, CA, USA: Aegean Park Press, 1984.

[18] L. R. Vermani, *Elements of Algebraic Coding Theory*, ser. Chapman Hall/CRC Mathematics Series. Taylor & Francis, 1996.

[19] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[20] Z. Li, L. Chen, L. Zeng, S. Lin and W. H. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Trans. on Communications*, vol. 54, no. 1, pp. 71–81, 2006.

[21] R. L. Townsend and E. J. Weldon, Jr., "Self-orthogonal quasi-cylic codes," *IEEE Trans. on Information Theory*, vol. IT-13, no. 2, pp. 183–195, 1967.

[22] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, no. 7, pp. 457–458, 1997.

[23] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, 2nd ed. San Francisco, CA: Kaufmann, 1988.

[24] T. J. Richardson and R. L. Urbanke, "The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 599–618, 2001.

[25] F. R. Kschischang, B. J. Frey and H.-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[26] V. Guruswami, "Iterative Decoding of Low-Density Parity Check Codes (An Introductory Survey)," Sept. 2006.

[27] S. J. Johnson, "Low-density parity-check codes from combinatorial designs," Ph.D. dissertation, School of Electrical Engineering and Computer Science, University of Newcastle, 2004.

[28] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. on Information Theory*, vol. 45, no. 2, pp. 399–431, 1999.

[29] M. P. C. Fossorier, M. Mihaljevic and H. Imai, "Reduced Complexity Iterative

Decoding of Low-Density Parity Check Codes Based on Belief Propagation," *IEEE Trans. on Communications*, vol. 47, no. 5, pp. 673–680, 1999.

[30] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier and X.-Y. Hu, "Reduced-Complexity Decoding of LDPC Codes," *IEEE Trans. on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.

[31] Y. Kou, S. Lin and M. P. C. Fossorier, "Low-density parity check codes based on finite geometries: a rediscovery," in *Proc. IEEE 2000 Int. Symp. Inform. Theory (ISIT)*, Sorrento, Italy, Jun. 2000, p. 200.

[32] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. on Information Theory*, vol. 47, no. 7, pp. 2711–2736, 2001.

[33] S. J. Johnson and S. R. Weller, "Resolvable 2-designs for low-density parity-check codes," *IEEE Trans. on Communications*, vol. 51, no. 9, pp. 1413–1419, 2003.

[34] B. Ammar, B. Honary, Y. Kou, J. Xu, and S. Lin, "Construction of low-density parity-check codes based on balanced incomplete block designs," *IEEE Trans. on Information Theory*, vol. 50, no. 6, pp. 1257–1268, 2004.

[35] B. Vasic and O. Milenkovic, "Combinatorial constructions of low-density parity check codes for iterative decoding," *IEEE Trans. on Communications*, vol. 50, no. 6, pp. 1156–1176, 2004.

[36] S. J. Johnson and S. R. Weller, "Codes for iterative decoding from partial geometries," *IEEE Trans. on Communications*, vol. 52, no. 2, pp. 236–243, 2004.

[37] P. O. Vontobel and R. M. Tanner, "Construction of codes based on finite generalized quadrangles for iterative decoding," in *Proc. IEEE Int. Symp. Information Theory*, 2001, p. 223.

[38] M. P. C. Fossorier, "Quasi-Cyclic Low-Density Parity-Check Codes From Circulant Permutation Matrices," *IEEE Trans. on Information Theory*, vol. 50, no. 8, pp.

1788–1793, 2004.

[39] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, 2nd ed. Cambridge, MA: MIT Press, 1972.

[40] J. Lu and J. M. F. Moura, "Structured LDPC codes for high-density recording: Large girth and low error floor," *IEEE Trans. on Magnetics*, vol. 42, no. 2, pp. 208–213, 2006.

[41] F. Zhang, Y. Xu, X. Mao, and W. Zhou, "High girth LDPC codes construction based on combinatorial design," in *IEEE 61st Vehicular Technology Conf.*, vol. 1, 2005, pp. 591–594.

[42] A. McGregor, "On the hardness of approximating stopping and trapping sets in LDPC codes," in *Proc. of the IEEE Information Theory Workshop (ITW 2007)*, Lake Tahoe, 2007, pp. 481–510.

[43] T. J. Richardson, "Error-floors of LDPC codes," in *Proc. 41st Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, October, 2003, pp. 1426–1435.

[44] S. Laendner and O. Milenkovic, "Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes," in *Proc. Conf. Wireless Communications*, Honolulu, HI, Jun. 2005, pp. 630–635.

[45] G. D. Forney, Jr., R. Koetter, F. R. Kschischang and A. Reznick, "On the effective weights of pseudocodewords for codes defined on graphs with cycles," in *Codes, Systems and Graphical Models*. New York: Springer-Verlag, 2001, pp. 101–112.

[46] R. Koetter and P. Vontobel, "Graph covers and iterative decoding of finite-length codes," in *Proc. 3rd Int. Symp. Turbo Codes and Related Topics*, Brest, France, Sept. 2003, pp. 75–82.

[47] C. Di, D. Proietti, E. Telatar, T. Richardson and R. Urbanke, "Finite length analysis of low-density parity-check codes," *IEEE Trans. on Information Theory*, vol. 48, no. 6, pp. 1570–1579, 2002.

[48] M. Schwartz and A. Vardy, "On the stopping distance and the stopping redundancy of codes," *IEEE Trans. on Information Theory*, vol. 52, no. 3, pp. 922–932, 2006.

[49] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles," *IEEE Trans. on Information Theory*, vol. 51, no. 3, pp. 929–953, 2005.

[50] N. Kashyap and A. Vardy, "Stopping sets in codes from designs," in *Proc. IEEE Int. Symp. Information Theory*, 2003, p. 122.

[51] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam and M. J. Wainwright, "Investigation of error floors of structured low-density parity-check codes by hardware emulation," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, 2006, pp. 1–6.

[52] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam and M. Wainwright, "Design of LDPC decoders for improved low error rate performance: Quantization and algorithm choices," *IEEE Trans. on Communications*, vol. 57, no. 11, pp. 3258–3268, 2009.

[53] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman and V. Stemann, "Practical loss-resilient codes," in *Proc. ACM Symp. on Theory of Computing*, 1997, pp. 150–159.

[54] J. Zhang, F.-W. Fu and D. Wan, "Stopping Sets of Algebraic Geometry Codes," *IEEE Trans. on Information Theory*, vol. 60, no. 3, pp. 1488–1495, 2014.

[55] K. Krishnan, K. Murali and L. S. Chandran, "Hardness of approximation results for the problem of finding the stopping distance in tanner graphs," in *FSTTCS 2006: Foundations of Software Technology and Theoretical Computer Science*, ser. Lecture Notes in Computer Science, S. Arun-Kumar and N. Garg, Ed. Springer Berlin Heidelberg, 2006, vol. 4337, pp. 69–80.

[56] D. J. C. MacKay and M. S. Postol, "Weaknesses of Margulis and Ramanujan-Margulis Low-Density Parity-Check Codes," in *Electronic Notes in Theoretical Computer Science*, vol. 74, Oct. 2003, pp. 97–104.

[57] D. V. Nguyen, S. K. Chilappagari, M. W. Marcellin and B. Vasic, "LDPC Codes from Latin Squares Free of Small Trapping Sets," *IEEE Trans. on Information Theory*, vol. 58, no. 4, pp. 2280–2302, 2012.

[58] M. Ivkovic, S. Chilappagari and B. Vasic, "Trapping sets in low-density parity-check codes by using Tanner graph covers," *IEEE Trans. on Information Theory*, vol. 54, no. 8, pp. 3763–3768, 2008.

[59] B. V. S. K. Chilappagari, D. V. Nguyen and M. W. Marcellin, "On Trapping Sets and Guaranteed Error Correction Capability of LDPC Codes and GLDPC Codes," *IEEE Trans. on Information Theory*, vol. 56, no. 4, pp. 1600–1611, 2010.

[60] M. Sipser and D. A. Spielman, "Expander Codes," *IEEE Trans. on Information Theory*, vol. 42, no. 6, pp. 1710–1722, 1996.

[61] S. K. Chilappagari, M. Chertkov, M. G. Stepanov and B. Vasic, "Instanton-based Techniques for Analysis and Reduction of Error Floors of LDPC Codes," *IEEE JSAC on Capacity Approaching Codes*, vol. 27, no. 6, pp. 855–865, 2009.

[62] L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic and M. J. Wainwright, "Predicting error floors of structured LDPC codes: Deterministic bounds and estimates," *IEEE J. Selected Areas Commun.*, vol. 27, no. 6, pp. 908–917, 2009.

[63] L. Dolecek, Z. Zhang, M. Wainwright, V. Anantharam and B. Nikolic, "Evaluation of the low frame error rate performance of LDPC codes using importance sampling," in *Proc. IEEE Information Theory Workshop*, Lake Tahoe, CA, Sept. 2007, pp. 202–207.

[64] L. Dolecek, J. Wang and Z. Zhang, "Towards improved LDPC code designs using absorbing set spectrum properties," in *Proc. 6th International Symposium on Tur-*

bo Codes and Iterative Information Processing (ISTC), 2010, pp. 477–481.

[65] J. Wang, L. Dolecek and R. Wesel, "Controlling LDPC Absorbing Sets via the Null Space of the Cycle Consistency Matrix," in *Proc. 2011 IEEE International Conference on Communications (ICC)*, Kyoto, Japan, Jun. 2011, pp. 1–6.

[66] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam and M. Wainwright, "Quantization effects in low-density parity-check decoders," in *Proc. IEEE Int. Conf. Communications*, Glasgow, Scotland, U.K., Jun. 2007, pp. 6231–6237.

[67] M. Yang, W.E. Ryan and Y. Li, "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Trans. on Communications*, vol. 52, no. 4, pp. 564–571, 2004.

[68] R. Echard and S.-C. Chang, "The π-rotation low-density parity-check codes," in *Proc. Global Telecommunications Conference*, San Antonio, TX, Nov. 2001, pp. 980–984.

[69] H. Jin, A. Khandekar and R. McEliece, "Irregular Repeat-Accumulate Codes," in *Proc. of the Second International Symposium on Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 1–8.

[70] S. J. Johnson and S. R. Weller, "Constructions for irregular repeat-accumulate codes," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Adelaide, Australia, Sept. 2005, pp. 179–183.

[71] ——, "Constructions for irregular repeat-accumulate codes," in *Proc. IEEE Int. Symp. Information Theory*, 2005, pp. 179–183.

[72] ——, "Interleaver and Accumulator Design for Systematic Repeat-Accumulate Codes," in *Proc. 6th Australian Communications Theory Workshop*, 2005, pp. 1–7.

[73] J. G. Michaels and K. H. Rosen, *Applications of Discrete Mathematics, Fourth Edition.* McGraw-Hill Inc., US, 1991.

[74] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson and R. L. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Letters*, vol. 5, no. 2, pp. 58–60, 2001.

[75] H. Zhang and J. M. F. Moura, "Large-girth LDPC codes based on graphical models," in *4th IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2003)*, Jun. 2003, pp. 100–104.

[76] S. J. Johnson and S. R. Weller, "High-rate LDPC codes from unital designs," in *IEEE Global Telecommunications Conference (GLOBECOM '03)*, vol. 4, Dec. 2003, pp. 2036–2040.

[77] N. Santhi and A. Vardy, "On the effect of parity-check weights in iterative decoding," in *Proc. IEEE Int. Symp. Information Theory*, Chicago, IL, 2004, p. 322.

[78] J. S. Yedidia, J. Chen, and M. Fossorier, "Generating code representations suitable for belief propagation decoding," in *Proc. 40st Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, October, 2002.

[79] C. J. Colbourn and J. H. Dinitz, *CRC Handbook of Combinatorial Designs*, 2nd ed. CRC Press, 2006.

[80] Th. Beth, D. Jungnickel, and H. Lenz, *Design Theory*, 2nd ed., ser. Encyclopedia of Math. and Its Applications, vol. 69/78. Cambridge University Press, 1999, vol. 1 and 2.

[81] M. Buratti, "On simple radical difference families," *J. Combin. Designs*, vol. 3, pp. 161–168, 1995.

[82] M. Genma, M. Mishima, and M. Jimbo, "Cyclic resolvability of cyclic Steiner 2-designs," *J. Combin. Designs*, vol. 5, pp. 177–187, 1997.

[83] E. F. Assmus, Jr. and J. D. Key, *Designs and their Codes*, ser. Cambridge Tracts in Math., vol. 103. Cambridge: Cambridge Univ. Press, 1992.

[84] J. Doyen, X. Hubaut and M. Vandensavel, "Ranks of incidence matrices of Steiner triple systems," *Math. Z.*, vol. 163, pp. 251–259, 1978.

[85] T. A. Gulliver, "Construction of quasi-cyclic quasi-cyclic codes," Ph.D. dissertation, Univserity of Victoria, Victoria, British Columbia, Canada, 1989.

[86] K. Chen, R. Wei and L. Zhu, "Existence of $(q, 7, 1)$ difference families with q a prime power," *J. Combin. Designs*, vol. 10, pp. 126–138, 2002.

[87] M. Greig and J. Abel, "Resolvable balanced incomplete block designs with block size 8," *Designs, Codes Cryptography*, vol. 11, pp. 123–140, 1997.

[88] M. Buratti, "On resolvable difference families," *Designs, Codes Cryptography*, vol. 11, pp. 11–23, 1997.

[89] C. Lam and Y. Miao, "On cyclically resolvable cyclic Steiner 2-designs," *J. Combin. Theory, Series A*, vol. 85, pp. 194–207, 1999.

[90] S. J. Johnson and S. R. Weller, "Interleaver and accumulator design for systematic repeat-accumulate codes," in *Proc. 6th Australian Communications Theory Workshop*, Brisbane, Australia, Feb. 2005.

[91] G. Liva, E. Paolini and M. Chiani, "Simple reconfigurable low-density parity-check codes," *IEEE Commun. Lett.*, vol. 9, no. 3, pp. 258–260, 2005.

[92] E. Netto, "Zur Theorie der Tripelsysteme," *Math. Ann.*, vol. 42, pp. 143–152, 1893.

[93] M. Buratti, "Constructions of $(q, k, 1)$ difference families with q a prime power and $k = 4, 5$," *Discrete Math.*, vol. 138, pp. 169–175, 1995.

[94] D. K. Ray-Chaudhuri and R. M. Wilson, "Solution of Kirkmans school-girl problem," in *Proc. Symp. Math.*, vol. 19, 1971, pp. 187–203.

[95] X.-Y. Hu, E. Eleftheriou and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. on Information Theory*, vol. 51, no. 1, pp. 386–398, 1964.

[96] S. Lin, H. Tang, Y. Kou, J. Xu and K. Abdel-Ghaffar, "Codes on finite geometries," in *Proc. IEEE Information Theory Workshop (ITW2002)*, Cairns, Australia, Sept. 2001, pp. 14–16.

[97] S. J. Johnson and S. R. Weller, "Regular low-density parity-check codes from combinatorial designs," in *Proc. IEEE Information Theory Workshop (ITW2002)*, Cairns, Australia, Sept. 2001, pp. 90–92.

[98] R. C. Bose and S. S. Shrikhande, "On the construction of sets of mutually orthogonal latin squares and the falsity of a conjecture of Euler," *Transactions of the American Mathematical Society*, vol. 95, pp. 191–209, 1960.

[99] R. M. Wilson, "Concerning the number of mutually orthogonal latin squares," *Discrete Mathematics*, vol. 9, pp. 181–198, 1974.

[100] R. C. Bose and S. S. Shrikhande, "On the construction of sets of mutually orthogonal latin squares and the falsity of a conjecture of Euler," *Transactions of the American Mathematical Society*, vol. 95, pp. 191–209, 1960.

[101] R. C. Bose, "On the Application of the Properties of Galois Fields to the Problem of Construction of Hyper-Græco-Latin Squares," *Sankhya: The Indian Journal of Statistics*, vol. 3, no. 4, pp. 323–338, 1938.

[102] A. C. H. Ling, C. J. Colbourn, M. J. Grannell and T. S. Griggs, "Construction Techniques for Anti-Pasch Steiner Triple Systems," *Journal of the London Mathematical Society*, vol. 2, no. 61, pp. 641–657, 2000.

[103] J. M. F. Moura, J. Lu and H. Zhan, "Structured low-density parity-check codes," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 42–55, 2004.

[104] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Efficient Erasure Correcting Codes," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 569–584, 2001.

[105] D. M. Donovan and M. J. Grannell, "On the number of transversal designs," *J.*

Combin. Theory, Series A, vol. 120, no. 7, pp. 1562–1574, 2013.

[106] G. B. Kyung and C.-C. Wang, "Exhaustive search for small fully absorbing sets and the corresponding low error-floor decoder," in *Proc. 2010 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2010, pp. 739–743.

[107] X. Zhang and P. H. Siegel, "Efficient Algorithms to Find All Small Error-Prone Substructures in LDPC Codes," in *Proc. 2011 IEEE Global Telecommunications Conference (GLOBECOM 2011)*, Dec. 2011, pp. 1–6.

[108] C. J. Colbourn and Y. Fujiwara, "Small stopping sets in Steiner triple systems," *Cryptogr. Commun.*, vol. 1, pp. 31–46, 2009.

[109] M. J. Grannell, T. S. Griggs and C. A. Whitehead, "The resolution of the anti-pasch conjecture," *J. Combin. Designs*, vol. 8, pp. 300–309, 2000.

[110] I. Diener, E. Schmitt and H. L. de Vries, "All 80 steiner triple systems on 15 elements are derived," *Discrete Mathematics*, vol. 55, no. 1, pp. 13–19, 1985.

[111] R. Mathon, K. T. Phelps and A. Rosa, "Small Steiner triple systems and their properties," *Ars Combin.*, vol. 15, pp. 3–11, 1983.

[112] P. Kaski and P. R. J. Östergård, "The Steiner triple systems of order 19," *Mathematics of Computation*, vol. 73, no. 248, pp. 2075–2092, 2004.

[113] Y. Fujiwara, "Even-freeness of cyclic 2-designs," e-print arXiv:1210.7516 [math.CO], 2012, available at: http://arxiv.org/abs/1210.7516.

[114] C. J. Colbourn, E. Mendelsohn, A. Rosa and J. Siran, "Anti-mitre steiner triple systems," *Graphs and Combinatorics*, vol. 10, no. 2-4, pp. 215–224, 1994.

[115] H. Tang, J. Xu, S. Lin and K. A. S. Abdel-Ghaffar, "Codes on finite geometries," *IEEE Trans. on Information Theory*, vol. 51, no. 2, pp. 572–596, 2005.

[116] Z. Liu and D. A. Pados, "Low complexity decoding of finite geometry LDPC codes," in *IEEE Int. Conf. on Comm. (ICC '03)*, vol. 4, May 2003, pp. 2713–

2717.

[117] J. Xu and L. Chen and I. Djurdjevic, S. Lin and K. A. S. Abdel-Ghaffar, "Construction of Regular and Irregular LDPC Codes: Geometry Decomposition and Masking," *IEEE Trans. on Information Theory*, vol. 53, no. 1, pp. 121–134, 2007.

[118] D. C. Clark, "Applications of finite geometries to designs and codes," Ph.D. dissertation, Michigan Technological University, 2011.

[119] Q. Huang and Q. Diao and S. Lin and K. A. S. Abdel-Ghaffar, "Cyclic and quasi-cyclic ldpc codes on constrained parity-check matrices and their trapping sets," *IEEE Trans. on Information Theory*, vol. 58, no. 5, pp. 2648–2671, 2012.

[120] W. Ryan and S. Lin, "Finite-Geometry LDPC Codes," in *Channel Codes*. Cambridge University Press, 2009, pp. 430–483.

[121] S. T. Xia and F. W. Fu, "On the stopping distance of finite geometry LDPC codes," *IEEE Comm. Letters*, vol. 10, no. 5, pp. 381–383, 2006.

[122] Y. Fujiwara, D. Clark, P. Vandendriessche, M. De Boeck and V. D. Tonchev, "Entanglement-assisted quantum low-density parity-check codes," *Physical Review A*, vol. 82, no. 4, 2010.

[123] N. Hamada, "The rank of the incidence matrix of points and d-flats in finite geometries," *J. Sci. Hiroshima Univ. Ser. A-I Math.*, vol. 32, no. 2, pp. 381–396, 1968.

[124] A. Frumkir and A. Yakir, "Rank of inclusion matrices and modular representation theory," *Israel J. Math.*, vol. 71, no. 3, pp. 309–320, 1990.

[125] N. Hamada, "On the p-rank of the incidence matrix of a balanced or partially balanced incomplete block design and its applications to error correcting codes,," *Hiroshima Math. J.*, vol. 3, no. 1, pp. 153–226, 1973.

[126] A. Yakir, "Inclusion matrix of k versus l affine subspaces and a permutation module

of the general affine group," *J. Combin. Theory Ser. A*, vol. 63, no. 2, pp. 301–317, 1993.

[127] P. Sziklai, "Polynomials in finite geometries," Feb. 2008.

[128] G. Van de Voorde, "On sets without tangents and exterior sets of a conic," 2012, e-print arXiv: 1201.0484v1 [math.CO], 2012, available at: http://arxiv.org/abs/1201.0484.

[129] N. J. Calkin, J. D. Key and M. J. de Resmini, "Minimum Weight and Dimension Formulas for Some Geometric Codes," *Des. Codes Cryptogr.*, vol. 17, no. 1–3, pp. 105–120, 1999.

[130] A. Blokhuis, A. Seress, H. A. Wilbrink, "On sets of points in $PG(2, q)$ without tangents," *Mitt. Math. Sem. Univ. Giessen*, pp. 39–44, 1991.

[131] M. Lavrauw, L. Storme, and G. Van de Voorde, "On the code generated by the incidence matrix of points and hyperplanes in $PG(n, q)$ and its dual," *Des. Codes Cryptogr.*, vol. 48, no. 3, pp. 231–245, 2008.

[132] B. F. Sherman, "Rédei Blocking Sets in Finite Desarguesian Planes," *J. Combin. Theory Ser. A*, vol. 98, no. 2, pp. 343–356, 2002.

[133] S. R. Weller and S. J. Johnson, "Regular low-density parity-check codes from oval designs," *European Transactions on Telecommunications*, vol. 14, no. 5, pp. 399–409, 2003.

[134] R. Mathon, "Constructions for Cyclic Steiner 2-designs," in *Combinatorial Design Theory*, ser. North-Holland Mathematics Studies, C. J. Colbourn and R. Mathon, Eds. North-Holland, 1987, vol. 149, pp. 353–362.

[135] M. J. Colbourn and R. A. Mathon, "On cyclic Steiner 2-designs," *Ann. Discrete Math.*, vol. 7, pp. 215–253, 1980.

[136] H. Hanani, "Balanced incomplete block designs and related designs," *Discrete*

Mathematics, vol. 11, no. 3, pp. 255–369, 1975.

[137] C. Reid and A. Rosa, "Steiner systems $S(2, 4, v)$ – a survey," *The Electronic Journal of Combinatorics*, 2010.

[138] J. Huang, L. Liu, W. Zhou and S. Zhou, "Large-Girth Nonbinary QC-LDPC Codes of Various Lengths," *IEEE Trans. on Communications*, vol. 58, no. 12, pp. 3436—3447, 2010.

[139] W. Sulek, M. Kucharczyk and G. Dziwoki, "Construction of Structured Nonbinary Low-Density Parity-Check Codes," *International Journal of e-Education, e-Business, e-Management and e-Learning*, vol. 3, no. 5, pp. 402–406, 2013.

[140] A. Anand and P. Senthilkumar, "A Novel Decoding Approach for Non-Binary LDPC Codes in Finite Fields," *International Journal of Electronics and Communication Engineering*, vol. 5, no. 2, pp. 133–141, 2012.