Edge Coloring and Stopping Sets Analysis in Product Codes with MDS components

Fanny Jardel, *Member, IEEE,* and Joseph J. Boutros, *Senior Member, IEEE,*

***Abstract*—We consider non-binary product codes with MDS components and their iterative row-column algebraic decoding on the erasure channel. Both independent and block erasures are considered in this paper. A compact graph representation is introduced on which we define double-diversity edge colorings via the rootcheck concept. An upper bound of the number of decoding iterations is given as a function of the graph size and the color palette size M. Then, we propose a differential evolution edge coloring algorithm that produces colorings with a large population of minimal rootcheck order symbols. The complexity of this algorithm per iteration is $o(M^{\aleph})$, for a given differential evolution parameter $\aleph$, where $M^{\aleph}$ itself is small with respect to the huge cardinality of the coloring ensemble. Stopping sets of a product code are defined in the context of MDS components and a relationship is established with the graph representation. A full characterization of these stopping sets is given up to a size $(d+1)^2$, where d is the minimum Hamming distance of the MDS component code. The performance of MDS-based product codes with and without double-diversity coloring is analyzed in presence of both block and independent erasures. In the latter case, ML and iterative decoding are proven to coincide at small channel erasure probability. Furthermore, numerical results show excellent performance in presence of unequal erasure probability due to double-diversity colorings.**

***Index Terms*—Product codes, MDS codes, iterative decoding, codes on graphs, differential evolution, distributive storage, edge coloring, diversity, erasure channel, stopping sets.**

I. Introduction

The colossal amount of data stored or conveyed by network nodes requires a special design of coding structures to protect information against loss or errors and to facilitate its access. At the end-user level, coding is essential for transmitting information towards the network whether it is located in a single node or distributed over many nodes. At the network level, coding should help nodes to reliably save a big amount of data and to efficiently communicate with each others. Powerful capacity-achieving error-correcting codes developed in the last two decades are mainly efficient at large or asymptotic block length, e.g. low-density parity-check (LDPC) codes [24] and their spatially-coupled ensembles [36], parallel-concatenated convolutional (Turbo) codes [6], [7], and polar codes derived from channel polarization [4]. Data transmission and storage in many nowadays networks may require short-length packets that are not suitable for capacity-achieving codes. The current interest in finite-length channel coding rates [45] put back the light on code design for short and moderate block length. Many potential candidates are available for this non-asymptotic length context such as binary and non-binary BCH codes, including Reed-Solomon (RS) codes, Reed-Muller (RM) codes, and tensor product codes of all these linear block codes [8], [40], [41].

Product codes, introduced by Peter Elias in 1954 [20], are tensor products of two (or more) simple codes with a structure that is well-suited to iterative decoding via its graphical description. In the early decades after their invention, product codes received a great attention due to their capability of correcting multiple burst errors [66], [72], the availability of erasure-error bounded-distance decoding algorithms [68], the ability of correcting many errors beyond the guaranteed correction capacity [1], and their efficient implementation with a variable rate [70]. The pioneering work by Tanner [62] brought new tools to coding theory and put codes on graphs, including product codes, and their iterative decoding in the heart of modern coding theory [33], [34], [51]. The graph approach of coding led to new optimal cycle codes on Ramanujan/Cayley graphs [63] and to generalizations of LDPC and product codes, known as GLD codes, studied for the binary symmetric channel (BSC) and the Gaussian channel [10]. The excellent performance of iterative (turbo) decoding of product codes on the Gaussian channel [47] made them compete with Turbo codes and LDPC codes for short and moderate block length. The convergence rate and stability of product codes iterative decoding were studied based on a geometric framework [57]. Product codes with mixed convolutional and block components were also found efficient in presence of impulsive noise [23]. In addition, iterated Reed-Muller product codes were shown to exhibit good decoding thresholds for the binary erasure channel, but at high and low coding rates only [67].

The class of product codes in which the row and the column code are both Reed-Solomon codes was extensively used since more than two decades in DVD storage media and in mobile cellular networks [71]. In these systems, the channel is modeled as a symbol-error channel without soft information, i.e. suited to algebraic decoding. Improvements were suggested for these RS-based product codes such as soft information provided by list decoding [54] within the iterative process in a Reddy-Robinson framework [50]. Also, RS-based product codes were directly decoded via a Guruswami-Sudan list decoder [29] after being generalized to bivariate polynomials [3]. For general tensor products of codes and interleaved

This manuscript was submitted to the IEEE Transactions on Information Theory, paper IT-15-1104, Dec. 2015. First revision completed on September 15, 2016 and second revision made on December 26, 2016. Part of the stopping sets analysis from this paper was presented at the IEEE International Symposium on Information Theory, Barcelona, July 2016. Fanny Jardel is with Nokia Bell-Labs, 70435 Stuttgart, Germany (email: fanny.jardel@nokia.com). She was with CEA, LIST, Gif Sur Yvette, F91191, France. Joseph J. Boutros is with the Dept. of Electrical and Computer Engineering, Texas A&M University at Qatar, Education City, 23874 Doha, Qatar (email: boutros@tamu.edu).

codes, a recent efficient list decoding algorithm was published [25], with an improved list size in the binary case. On channels with soft information, RS-based product codes may be row-column decoded with soft-decision constituent decoders [21], [31].

Tolhuizen found the Hamming weight distribution of both binary and non-binary product codes up to a weight less than $d_1 d_2 + \max(d_1\lceil d_2/q\rceil, d_2\lceil d_1/q\rceil)$ [64], where d_1, d_2 are the minimum Hamming distances of the component codes and q is the finite field size. Enumeration of erasure patterns up to a weight less than $d_1 d_2 + \min(d_1, d_2)$ was realized by Sendrier for product codes with MDS components [58]. Rosnes studied stopping sets of binary product codes under iterative ML-component-wise decoding [53], where the defined stopping sets and their analysis are based on the generalized Hamming distance [30], [69].

A. Paper content and structure

In this paper, we consider non-binary product codes with MDS components and their iterative algebraic decoding on the erasure channel. Both independent and block erasures are considered in our paper. The erasure channel is currently a major area of research in coding theory [37], [38] because of strong connections with theoretical computer science [38] and its model that easily allows to understand the behavior of codes such as for LDPC codes [18], for general linear block codes [56], and for turbo codes [52]. Coding for block erasures was examined by Lapidoth in the context of convolutional codes [39]. This was a basis to later construct codes for the block-fading channel with additive white Gaussian noise [14], [28]. The notion of *rootcheck* introduced in [13], [14] for single-parity checknodes was applied to more general checknodes in GLD codes [12] and product codes [11] to achieve diversity on non-ergodic block-fading channels. The rootcheck concept is the main tool in this paper, in a way similar to [11], to define a compact graph representation and study iterative decoding in presence of block erasures. Edge coloring is one of the most interesting problems in modern graph theory [9]. In this paper, edge coloring is a tool, when combined to the rootcheck concept, yields double-diversity product codes. Our work is valid for finite-length MDS-based product codes only. Product codes for asymptotic block length were studied for single-parity codes constituents [48] and for the erasure channel with a standard regular structure [55] and MDS-based irregular structures [2].

Whether a product code is endowed with an edge coloring or not, the analysis of stopping sets, their characterization and their enumeration is a fundamental task to be able to design codes for erasure channels and determine the decoder performance. Our work in this sense is an improvement to previous works cited above by Tolhuizen, Sendrier, and Rosnes. Besides this objective of stopping sets characterization which is useful for independent channel erasures and erasures occurring in blocks of symbols, recent works on locality [26] stimulated us to search for edge colorings with a large population of edges that admit a minimal rootcheck order. Locality is a concept encountered in distributive storage [35], [49] where classic coding theory is adapted to the nature of a network with distributed nodes with its own constraints of load in bandwidth and storage [19], [43]. Furthermore, product codes with MDS components appear to be suited to distributive storage [22] owing to their simple and mature techniques of erasure resilience. In our search for good edge colorings, we provide a new algorithm based on the concept of differential evolution [44], [61]. Our MDS-based product codes equipped with a double-diversity edge coloring are suited to distributed storage applications and to wireless networks where diversity is a key parameter. MDS-based product codes do not exhibit the best locality for multiple erasures as locally repairable codes codes with sequential recovery [5], nonetheless they can fill block erasures thanks to double diversity which is an advantage over locally repairable codes.

The paper is structured as follows. Section II gives a list of mathematical notations. The graph representation of product codes is given in Section III, including compact and non-compact graphs. The rootcheck concept and its consequences are also found in Section III. Our edge coloring algorithm for bipartite graphs of product codes is described in Section IV. The analysis of stopping sets is made in Section V. Finally, in Section VI, we study the performance of product codes with MDS components on erasure channels and we give theoretical and numerical results before the conclusions in the last section.

B. Main results

The main results in this paper are:

- Establishing a new compact graph for product codes. The compact graph has many advantages, the main one being its ability to imitate a Tanner graph with parity-check nodes. The compact graph is also the basis for the differential evolution edge coloring. See Section III-B.
- Iterative decoding analysis of finite-length product codes, mainly the proof of new bounds on the number of decoding iterations. See Theorem 1 and Corollary 1.
- A new edge coloring algorithm (DECA) capable of producing double-diversity colorings despite the huge size of the coloring ensembles. See Section IV-B. Construction via the DECA algorithm of product codes maximizing the number of edges with root order 1, i.e. minimizing the locality when the process of repairing nodes is considered. See Section IV-C.
- Proving new properties of stopping sets for product codes with MDS components. See Propositions 1&2, Corollaries 2-4, and Lemmas 2&3.
- Complete enumeration and characterization of stopping sets up to a size $(d_1+1)(d_2+1)$. This stopping set enumeration goes beyond the weight $d_1 d_2 + \max(d_1, d_2)$ of Tolhuizen's Theorem 3 for codeword enumeration in the MDS components case. See Lemmas 4&5 and Theorem 2. The enumeration theorem for $d_1 \neq d_2$ is not included in this paper due to the lack of space. It can be found in a longer paper on arXiv.
- First numerical results for MDS-based product codes on erasure channels showing how close iterative decoding is to ML decoding, mainly for small ϵ. We proved that

iterative decoding perform as well as ML decoding (the ratio of error probabilities tends to 1) for MDS-based product codes at small ϵ. See Proposition 3, Corollary 5, and other performance results in Section VI-B.
- Great advantage of double-diversity colorings of product codes (with respect to codes without coloring) in presence of unequal probability erasures. Thus, double-diversity colorings are efficient on both ergodic and non-ergodic erasure channels. See Section VI-C.

II. Mathematical notation and Terminology

We start by the notation related to the product code and its row and column components. The impatient reader may skip this entire section and then refer to it later to clarify any notation within the text. Basic notions on product codes and fundamental properties are found in main textbooks [8], [40], [41] and the encyclopedia of telecommunications [33].
The column code C_1 is a linear block code over the finite field $\mathbb{F}_q$ with parameters $[n_1, k_1, d_1]_q$ which may be summarized by $[n_1, k_1]$ when no confusion is possible. The integer q is the code alphabet size, n_1 is the code length, k_1 is the code dimension as a vector subspace of $\mathbb{F}_q^{n_1}$, and d_1 is the minimum Hamming distance of C_1. Similarly, the row code C_2 is a linear block code with parameters $[n_2, k_2, d_2]_q$. Let G_1 and G_2 be two matrices of size $k_1 \times n_1$ and $k_2 \times n_2$ containing in their row a basis for the subspaces C_1 and C_2 respectively. From the two generator matrices G_1 and G_2 a product code C_P is constructed as a subspace of $\mathbb{F}_q^N$ with a generator matrix $G_P = G_1 \otimes G_2$, where $N = n_1 n_2$ and $\otimes$ denotes the Kronecker product [41]. C_P has dimension $K = k_1 k_2$ and minimum Hamming distance $d_P = d_1 d_2$. C_1 and C_2 are also called component codes, this is a terminology from concatenated codes. In [62] and [11], vertices associated to component codes are called subcode nodes.

A linear $[n, k, d]_q$ code is said to be MDS, i.e. Maximum Distance Separable, if it satisfies $d = n - k + 1$. Binary MDS codes are the trivial repetition codes and the single parity-check codes. In this paper, we only consider non-trivial non-binary MDS codes where $q > n > 2$. A linear code over $\mathbb{F}_q$ of rate $R = k/n$ is said to be MDS diversity-wise or MDS in the block-fading/block-erasure sense if it achieves a diversity order L such that $L = 1 + \lfloor M(1-R) \rfloor$, where M is the number of degrees of freedom in the channel. The right term $1 + \lfloor M(1-R) \rfloor$ is known as the block-fading Singleton bound [32], [42]. In this paper, M shall denote the number of colors, i.e. the palette size of an edge coloring. Assume that code symbols are partitioned into M sub-blocks, a code is said to attain diversity L if it is capable of correct decoding when $L-1$ sub-blocks are erased by the channel. The reader should refer to [65], chapter 3, for an exact definition of diversity on fading channels with additive white Gaussian noise.

A product code shall be represented by a non-compact graph $\mathcal{G} = (V_1, V_2, E)$. $\mathcal{G}$ is a complete bipartite graph where V_1 is the set of n_2 right vertices, V_2 is the set of n_1 left vertices, and E is the set of N edges representing the code symbols. A compact graph $\mathcal{G}^c$ will also be introduced in the next section with $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$. The number of edges (also called super-edges) in the compact graph is $|E^c| = N^c$. A super-edge is equivalent to a super-symbol that represents $(\ n_1 - k_1\ ) \times (\ n_2 - k_2\ )$ symbols from $\mathbb{F}_q$. The ensemble of edge colorings is denoted $\Phi(E)$ and $\Phi(E^c)$ for $\mathcal{G}$ and $\mathcal{G}^c$ respectively. An edge coloring will be denoted by ϕ.

Under iterative row-column decoding, the rootcheck order ρ is equal to the number of decoding iterations required to solve the edge value (or the symbol associated to that edge). In this paper, one decoding iteration is equivalent to decoding all rows or decoding all columns. A sequence of n_1 row decoders followed by a sequence of n_2 column decoders is counted as two decoding iterations. Given ϕ, the rootcheck order of an edge is $\rho(e)$. The greatest $\rho(e)$ among all edges will be referred to as $\rho_{max}(\phi)$. The number of edges e satisfying $\rho(e) = 1$ is $\eta(\phi)$, this is the number of good edges and will be processed by the DECA algorithm in Section IV. The DECA parameter $\aleph$ shall represent the number of edges to be mutated, i.e. those edges being chosen in the population of bad edges satisfying $\rho(e) > 1$.

We give now a general definition of a stopping set. A detailed study is found in Section V. The notion of a stopping set is useful for iterative decoding in presence of erasures [18].

Definition 1: Let $C[n, k]_q$ be a linear code and $\mathcal{D}$ be a decoder based on a deterministic decoding method. Consider a set $\mathcal{S}$ of s fixed positions $i_1, i_2, \ldots, i_s$, where $1 \leq i_j \leq n$, and assume all code symbols on the s positions given by $\mathcal{S}$ are erased. The set $\mathcal{S}$ is said to be a Stopping Set if $\mathcal{D}$ fails in retrieving none of the s erased symbols.

This paper focuses on stopping sets of a product code under iterative algebraic row-column decoding, i.e. referred to as type II stopping sets. The number of stopping sets of size w is τ_w. The rectangular support $\mathcal{R}(\mathcal{S})$ of a stopping set $\mathcal{S}$ can be seen as the smallest rectangle containing $\mathcal{S}$ inside the $n_1 \times n_2$ product code rectangular representation. After excluding rows and columns not involved in $\mathcal{S}$, the rectangular support has size $\ell_1 \times \ell_2$ where $w = |\mathcal{S}| \leq \ell_1 \ell_2$. The word error performance of C_P shall be estimated on erasure channels, P_{ew}^{ML} is the word error probability under Maximum Likelihood decoding and $P_{ew}^{\mathcal{G}}$ is the word error probability under iterative row-column decoding. Three erasure channels are considered: 1- The Symbol Erasure Channel, $SEC(q, \epsilon)$, where code symbols are independently erased with a probability ϵ, 2- The Color Erasure Channel, $CEC(q, \epsilon)$, where all symbols associated to the same color are block-erased with a probability ϵ. On the $CEC(q, \epsilon)$, block-erasure events are independent from one color to another. 3- The unequal probability Symbol Erasure Channel, $SEC(q, \{\epsilon_i\}_{i=1}^{M})$, where symbol erasures are independent but their erasure probability varies from one color to another.

III. Graph representations for diversity

Efficient graph representation of codes was established by Tanner for different types of coding structures [62]. Bounds on the code parameters and iterative decoding algorithms were also proposed for codes on graphs [62]. In this paper, we study the edge coloring of a product code graph, where edges represent code symbols. As shown below, the original graph for a product code is too complex, i.e. it leads to a large ensemble of colorings. Hence, we introduce a compact graph where symbols are grouped together with the same color in order to reduce the size of the coloring ensemble. The compact graph also has another asset: grouping parity symbols together renders check nodes similar to parity-check nodes found in standard low-density parity-check codes [24] [51].

A. Non-compact graph

Consider a product code $C_1[n_1, k_1]_q \otimes C_2[n_2, k_2]_q$ where C_1 is the column code and C_2 is the row code. The product code is defined over the finite field $\mathbb{F}_q$ and has length N and dimension K given by [41]

$$N = n_1 n_2, \qquad K = k_1 k_2. \tag{1}$$

Each code symbol simultaneously belongs to one row and to one column. Product codes studied in this paper are regular, in the sense that all columns are codewords of C_1 and all rows are codewords of C_2. The graph of $C_1[n_1, k_1]_q \otimes C_2[n_2, k_2]_q$ is built as follows. We use the same terminology as in [51]:

- n_1 check nodes are drawn on the left. A left check node represents the coding constraint which states that a row belongs to C_2. The n_1 left check nodes are referred to as C_2 check nodes, or row check nodes, or equivalently left vertices.
- n_2 check nodes are drawn on the right. A right check node represents the coding constraint which states that a column belongs to C_1. The n_2 right check nodes are referred to as C_1 check nodes, or column check nodes, or equivalently right vertices.
- An edge is drawn between a left vertex and right vertex. It represents a code symbol located on the row of the left vertex and on the column of the right vertex. The code symbol belongs to $\mathbb{F}_q$.

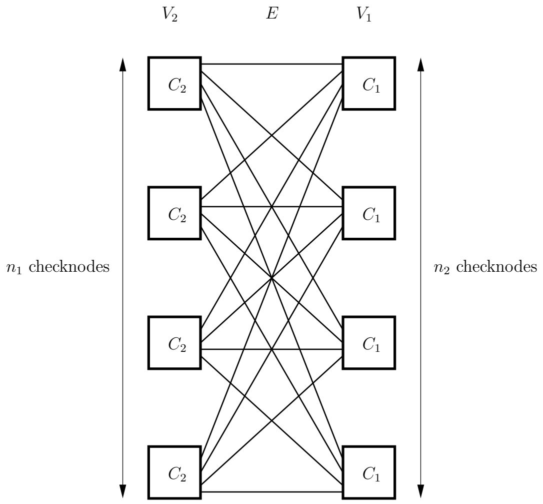


Figure 1: Non-compact bipartite graph $\mathcal{G} = (V_1, V_2, E)$ of a product code $[4,2]^{\otimes 2}$, i.e. $n_1 = n_2 = 4$, $k_1 = k_2 = 2$, $|V_1| = |V_2| = 4$, and $|E| = N = n_1 n_2 = 16$ edges representing 16 symbols in $\mathbb{F}_q$.

In summary, the product code graph (V_1, V_2, E) is a complete biregular bipartite graph built from n_1 left vertices, n_2 right vertices, and $N = |E| = n_1 n_2$ edges representing code symbols. The left degree is n_2 and the right degree is n_1. Irregular product codes can be found in [2]. Our paper is restricted to regular product codes. Figure 1 shows the bipartite graph of a square regular symmetric product code $[4,2] \otimes [4,2]$. The graph structure reveals n_1, n_2, and $N = n_1 n_2$. The dimensions k_1 and k_2 of the component codes have no effect on the number of vertices and edges in the product code graph. Indeed, a $[4,3] \otimes [4,3]$ code can also be defined by the graph in Figure 1. The role of the dimensions k_1 and k_2 is played within the check constraints inside left and right vertices. Similarly, the size of the finite field defining the code cannot be revealed from the graph structure, i.e. the product code graph does not depend on q.

Definition 2: The non-compact graph $\mathcal{G} = (V_1, V_2, E)$ for a $[n_1, k_1] \otimes [n_2, k_2]$ product code is a complete bipartite graph with $n_1 = |V_2|$ left vertices and $n_2 = |V_1|$ right vertices.

B. Compact graph

In [11] where the diversity of binary product codes was considered, vertices of the non-compact graph were grouped together into super-vertices (or supernodes) because the different channel states lead to multiple classes of check nodes as in root-LDPC codes [14]. To render a graph-encodable code, supernodes in [11] were made by putting $n-k$ nodes together for a $[n,k]$ component code. Also, $n-k$ is not necessarily a divisor of n.

Definition 3: The compact graph $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$ for a $[n_1, k_1] \otimes [n_2, k_2]$ product code is a complete bipartite graph with $\lceil \frac{n_1}{n_1 - k_1} \rceil = |V_2^c|$ left vertices and $\lceil \frac{n_2}{n_2 - k_2} \rceil = |V_1^c|$ right vertices.

From the above definition, the number of edges in the compact graph $\mathcal{G}^c$ is found to be

$$N^c = |E^c| = \left\lceil \frac{n_1}{n_1 - k_1} \right\rceil \times \left\lceil \frac{n_2}{n_2 - k_2} \right\rceil . \tag{2}$$

Assuming that $(n_1 - k_1)$ divides n_1 and $(n_2 - k_2)$ divides n_2, a left check node in $\mathcal{G}^c$ is equivalent to $n_1 - k_1$ row constraints and a right check node in $\mathcal{G}^c$ is equivalent to $n_2 - k_2$ column constraints. An edge in the compact graph carries $(n_1 - k_1) \times (n_2 - k_2)$ code symbols. To avoid confusion between edges of $\mathcal{G}$ and $\mathcal{G}^c$, we may refer to those in $\mathcal{G}^c$ as super-edges or equivalently as super-symbols. If n_i is not multiple of $n_i - k_i$, then the last row or column supernode will contain less than $n_i - k_i$ check nodes. Figure 2 depicts the compact graph of the $[4,2]^{\otimes 2}$ product code. All $[n, n/2]^{\otimes 2}$ product codes have a compact graph identical to that of $[4,2]^{\otimes 2}$, for all $n \geq 2$, n even.

C. Diversity and codes on graphs

From a coding point of view, diversity is the art of creating many replicas of the same information. From a channel

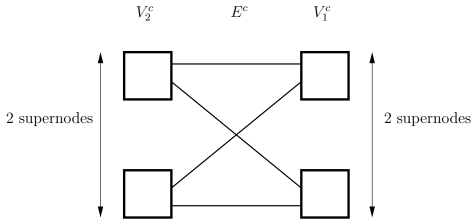


Figure 2: Compact bipartite graph $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$ with two supernodes on each side for the product code $[n, n/2]^{\otimes 2}$, $|V_1^c| = |V_2^c| = 2$ and $|E^c| = N^c = 4$ supersymbols. Each super-symbol (i.e. super-edge) contains $n^2/4$ symbols (i.e. edges).

point of view, diversity is the number of degrees of freedom available while transmitting information. In distributive storage, independent failure of individual machines is modeled by independent erasures of code symbols, while the outage of a cluster of machines is modeled as block erasures of code symbols. Assuming a storage domain with a large set of machines partitioned into M clusters, diversity of distributed coding is defined as follows:

Definition 4: Consider a product code C_P defined over $\mathbb{F}_q$. Assume that symbols are given M different colors. Erasing one color is equivalent to erasing all symbols having this color. The code is said to achieve a diversity L if it is capable of filling all erasures after erasing $L-1$ colors. The code is full-diversity when $L = M$.

The integer L may also be called the diversity order. For Gaussian channels with fading, the diversity order appears as the slope of the error probability at high signal-to-noise ratio γ, i.e. $L = \lim_{\gamma\to\infty} -\frac{\log P_e}{\log \gamma}$ [14]. In the above definition, a cluster has been replaced by a color. We will use this terminology throughout the paper. Notice that coloring symbols is equivalent to edge coloring of the product code graph. The number of edges is N in the non-compact graph and N^c in the compact graph. In the sequel, all colorings are supposed to be perfectly balanced, i.e. M divides both N and N^c and the number of edges having the same color is N/M and N^c/M for the non-compact graph and the compact graph respectively. More formally, our edge coloring is defined as follows: an edge coloring ϕ of $\mathcal{G} = (V_1, V_2, E)$ is a mapping associating one color to every edge in E,

$$\phi : E \to \{1, 2, \ldots, M\}, \tag{3}$$

such that $|\phi^{-1}(i)| = N/M$ for $i = 1 \ldots M$, where $\phi^{-1}(i)$ is the inverse image of i. Similarly, $\phi : E^c \to \{1, 2, \ldots, M\}$ for $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$ and $|\phi^{-1}(i)| = N^c/M$. The set of such mappings for $\mathcal{G}$ and $\mathcal{G}^c$ is denoted $\Phi(E)$ and $\Phi(E^c)$ respectively.

Consider a coloring ϕ in $\Phi(E^c)$. It can be embedded into $\Phi(E)$ by copying the color of a super-edge to its associated $(n_1 - k_1) \times (n_2 - k_2)$ edges in E. Thus, let $\Phi(E^c \to E)$ be the subset of colorings in $\Phi(E)$ obtained by embedding all colorings of $\Phi(E^c)$ into $\Phi(E)$. We have

$$\Phi(E^c \to E) \subset \Phi(E) \quad \text{and} \quad |\Phi(E^c \to E)| = |\Phi(E^c)|. \tag{4}$$

The size of the edge coloring ensembles $\Phi(E)$ and $\Phi(E^c)$ is obviously not the same when $N^c < N$, which occurs for both row and column component codes not equal to single parity-check codes. Indeed, when a palette of size M is used to color edges, the total number of colorings of E is

$$|\Phi(E)| = \frac{N!}{((N/M)!)^M}. \tag{5}$$

This number for the compact graph is

$$|\Phi(E^c)| = \frac{N^c!}{((N^c/M)!)^M}. \tag{6}$$

As an example, for the $[12, 10]^{\otimes 2}$ code and $M = 4$, there are $2 \cdot 10^{83}$ edge colorings for the non-compact graph and $2 \cdot 10^{19}$ edge colorings for the compact graph. It is clear that the construction of product codes for diversity is much easier when based on $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$ because its edge coloring ensemble is smaller. Furthermore, as described below, vertices in $\mathcal{G}^c$ act in a way similar to standard LDPC check nodes making the design very simple. Furthermore, we will see in Section V that edge colorings of the compact graph render larger stopping sets than colorings of the non-compact graph.

The diversity order L attained by a code can never exceed M, the latter being the diversity from a channel point of view. A tighter upper bound of L showing the rate-diversity tradeoff is the block-fading Singleton bound. The Singleton bound for the maximal achievable diversity order is valid for all types of non-ergodic channels, including block-erasure and block-fading channels. The block-fading Singleton bound states that [32] [42]

$$L \le 1 + \lfloor M(1 - R) \rfloor, \tag{7}$$

where $R = K/N$ is the coding rate of the product code. Codes satisfying the equality in the above Singleton bound are referred to as diversity-wise MDS or block-fading MDS codes. From (7), we deduce that $R \le 1/M$ if $L = M$ (full-diversity coding). For example, we get $R \le 1/2$ with an edge coloring using $L = M = 2$ colors and $R \le 1/4$ for $L = M = 4$ colors. The coding rate can exceed $1/M$ when $L < M$ in applications where full diversity is not mandatory. An example suited to distributed storage is an edge coloring with a palette of $M = 4$ colors, a diversity $L = 2$, and $R \le 3/4$.

D. Rootcheck nodes and root symbols

In a way similar to root-LDPC codes and product codes built for block-fading channels [11], [14], we introduce now the notion of root symbols and root-check nodes in product codes to be designed for distributive storage. A linear $[n, k]_q$ code with parity-check matrix H can fill $n-k$ erasures at positions where the columns of H are independent. These $n-k$ symbols correspond to $n-k$ separate edges in the non-compact graph and to a unique edge (supersymbol) in the compact graph. Therefore, for simplicity, we start by defining a root supersymbol in the compact graph where supernodes are equivalent to standard LDPC parity-check nodes.

Definition 5: Let $\mathcal{G}^c$ be a compact graph of a product code, let ϕ be a given edge coloring, and let $e \in E^c$ be a supersymbol. e is a *root supersymbol* with respect to $\phi(e)$ if it admits a neighbor vertex υ, $\upsilon \in V_1^c$ or $\upsilon \in V_2^c$, such that all adjacent edges f in υ satisfy $\phi(f) \neq \phi(e)$.

In Definition 5, if $\upsilon \in V_1^c$ then e is a root supersymbol thanks to the product code column to which it belongs, i.e. e can be solved in one iteration by its column component code when the color $\phi(e)$ is erased. Likewise, e is protected against erasures by its row component code if $\upsilon \in V_2^c$ in the previous definition. Finally, a root supersymbol may be doubly protected by both its row and its column if both right and left neighbors $\upsilon_1 \in V_1^c$ and $\upsilon_2 \in V_2^c$ satisfy the condition of Definition 5.

Definition 6: Let $\mathcal{G}$ be a non-compact graph of a product code, let ϕ be a given edge coloring, and let $e \in E$ be a symbol. e is a *root symbol* with respect to $\phi(e)$ if it admits a neighbor vertex υ such that:
$\phi(f) = \phi(e)$ for at most $n_2 - k_2 - 1$ adjacent edges f if $\upsilon \in V_1$, or
$\phi(f) = \phi(e)$ for at most $n_1 - k_1 - 1$ adjacent edges f if $\upsilon \in V_2$.

As mentioned in the paragraph before Definition 5, Definition 6 implies that the $n_i - k_i$ root symbols with the same color should belong to positions of independent columns in the parity-check matrix of the component code C_i. This constraint automatically disappears for MDS component codes since any set of $n_i - k_i$ columns of H_i has full rank.

E. The rootcheck order in product codes

Not all symbols of a product code are root symbols. Under iterative row-column decoding on channels with block erasures, some symbols may be solved in two decoding iterations or more. Some set of symbols may never be solved and are referred to as stopping sets [18], [53], [56]. Our study is restricted to erasing the symbols of one color out of M. Hence, the rest of this paper is restricted to double diversity, $L = 2$. Absence of diversity is equivalent to $L = 1$. We establish now the root order ρ of a symbol. For root symbols satisfying Definitions 5 and 6, the root order is $\rho = 1$. For symbols that can be solved after two decoding iterations, we set $\rho = 2$. The formal definition of the root order $\rho(e)$ of an edge e (or a super-symbol e) can be written in a recursive manner as in Definition 7 below. For all super-symbols $e \in E^c$, start by setting $\rho(e) = \infty$.

Definition 7: Let $\mathcal{G}^c$ be a compact graph of a product code and let $\phi \in \Phi(E^c)$ be an edge coloring. The super-symbol e has *root order* $\rho(e) = \min(\rho_1, \rho_2)$ where:
1- Let $\upsilon_1 \in V_1^c$ be the column neighbor vertex of e. If υ_1 contains no edge f such that $\phi(f) = \phi(e)$ then set $\rho_1 = 1$. Otherwise, set ρ_1 to the smallest integer such that $\forall f$ adjacent to e in υ_1 and $\phi(f) = \phi(e)$, we have $\rho(f) < \rho_1$. Take $\rho_1 = \infty$ if a finite integer cannot be found.
2- Let $\upsilon_2 \in V_2^c$ be the row neighbor vertex of e. If υ_2 contains no edge f such that $\phi(f) = \phi(e)$ then set $\rho_2 = 1$. Otherwise, set ρ_2 to the smallest integer such that $\forall f$ adjacent to e in υ_2 and $\phi(f) = \phi(e)$, we have $\rho(f) < \rho_2$. Take $\rho_2 = \infty$ if a finite integer cannot be found.

The previous definition implies that $\rho(e) = 1$ if there exists no adjacent edge with the same color. When color $\phi(e)$ is erased, symbols belonging to the so-called stopping sets can never be solved (even after an infinite number of decoding iterations) and hence their root order ρ is infinite. In Section V we review stopping sets as known in the literature and we study new stopping sets for product codes based on MDS components under iterative algebraic decoding. Definition 7 can be rephrased to make it suitable for the non-compact graph $\mathcal{G}$. We pursue this section to establish an upper bound of the largest finite root order valid for all edge colorings ϕ.

Theorem 1: Let C_P be a product code $[n_1, k_1] \otimes [n_2, k_2]$ with a compact graph $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$. $\forall \phi \in \Phi(E^c)$ and $\forall e \in E^c$ we have:
Case 1: $\nexists f \in E^c$ such that $\phi(f) = \phi(e)$ and $\rho(f) = \infty$, then

$$1 \leq \rho(e) \leq \left\lceil \frac{N^c}{2M} \right\rceil = \rho_u.$$

Define the minimum number of good edges,

$$\eta_{min}(\phi) = \min_{i=1\ldots M} |\{f \in E^c : \phi(f) = i, \rho(f) = 1\}|.$$

Then, in Case 1,

$$2\rho(e) + \eta_{min}(\phi) - 3 \leq \left\lceil \frac{N^c}{M} \right\rceil. \qquad (8)$$

Case 2: $\exists f \in E^c$ such that $\phi(f) = \phi(e)$ and $\rho(f) = \infty$, then

$$\rho(e) = \infty \quad or \quad 1 \leq \rho(e) \leq \left\lceil \frac{N^c}{M} \right\rceil - 4,$$

where $N^c = |E^c|$ is given by (2).

Proof: Case 1 corresponds to a product code with diversity $L = 2$, for a given color $\phi(e)$, which is capable of solving all symbols when that color is erased. The graph has no infinite root order symbols. ρ is recursively built by starting from $\rho = 1$ following two paths in the graph until reaching a common edge e that has two neighboring vertices with edges of order $\rho(e) - 1$. There are up to $\lceil N^c/M \rceil$ edges, including e, having color equal to $\phi(e)$. The largest $\rho(e)$ is attained in the middle of the longest path of length $\lceil N^c/M \rceil$, hence $2\rho(e) - 1 \leq \lceil N^c/M \rceil$ which is translated into the stated result for Case 1. An illustrated instance is given for the reader in Example 1. Back to the path of length $2\rho(e) - 1$ ending with edges of order 1 on both sides, if the population of order 1 edges is η_1 for the color $\phi(e)$, then the path can only use a maximum of $\lceil N^c/M \rceil - (\eta_1 - 2)$ edges. We get the inequality $2\rho(e) - 1 \leq \lceil N^c/M \rceil - (\eta_1 - 2)$. By plugging $\eta_{min}(\phi)$ instead of η_1, this inequality becomes independent from the particular color. The stated inequality in (8) is obtained after grouping $\rho(e)$ and $\eta_{min}(\phi)$ on the left side.

Case 2 corresponds to bad edge coloring where the product code does not have double diversity, i.e. stopping sets do exist for the color $\phi(e)$. The order of e may be infinite if e is involved in a stopping set with another edge f having the same color. Otherwise, consider the smallest stopping set of size four symbols (the smallest cycle in $\mathcal{G}^c$ with edges of color $\phi(e)$), then there remains $\lceil N^c/M \rceil - 4$ edges of color $\phi(e)$. A path of length $\lceil N^c/M \rceil - 4$ starting with $\rho = 1$ and ending at $\rho = \infty$ may exist. The largest finite order in this path before reaching the stopping set is $\rho = \lceil N^c/M \rceil - 4$. ■

Corollary 1: Let C_P be a product code $[n_1, k_1] \otimes [n_2, k_2]$ with a compact graph $\mathcal{G}^c$. Let $\phi \in \Phi(E^c)$ be an edge coloring. We define

$$\rho_{max}(\phi) = \max_{e \in E^c} \rho(e). \tag{9}$$

C_P attains double diversity under iterative row-column decoding if and only if $\rho_{max}(\phi) < \infty$. In this case, we say that ϕ is a double-diversity coloring and $\forall e \in E^c$, e can be solved after at most ρ_{max} decoding iterations where $\rho_{max}(\phi) \leq \rho_u$.

For colorings in $\Phi(E)$, we extend the same definition as in Corollary 1 and we say that $\phi \in \Phi(E)$ is double-diversity if all edges have a finite rootcheck order. The parameter ρ_{max} is important in practical applications to bound from above the amount of conveyed information within a network (whether it is a local-area or a wide-area network). In fact, in coding for distributed storage, the locality of a product code per decoding iteration is $\max(n_1 - 1, n_2 - 1)$ in $\mathcal{G}$ under algebraic decoding of its row and column components. Here, the locality is the number of symbols to be accessed in order to repair an erased symbol [26]. Locality is $\max(k_1, k_2)$ for MDS components under ML decoding of the product code components. Finally, for a product code, the information transfer per symbol is bounded from above by

$$\rho_{max}(\phi) \times \max(n_1 - 1, n_2 - 1). \tag{10}$$

The exact transfer cost to fill all erasures with iterative decoding can be determined by multiplying each order ρ with the corresponding edge population size. This exact cost may vary in a wide range from one coloring to another. The DECA algorithm presented in Section IV enlarges the edge population with root order 1 leading in practice to a dramatic reduction of ρ_{max}. The interdependence between ρ and the population of order 1 was revealed in inequality (8). This inequality is useful in intermediate cases where $\rho_{max} = 1$ is not attained, i.e. outside the case where all edges have order 1. The influence of the component decoding method on the performance of a product code via its stopping sets is discussed in Section V.

Example 1: Consider a $[12, 10]^{\otimes 2}$ product code and a coloring ϕ with $M = 4$ colors. The compact graph has $|E^c| = 6 \times 6$ edges. Instead of drawing $\mathcal{G}^c$, we draw the 6×6 compact matrix representation of the product code in Fig. 3. Supersymbols corresponding to a color $\phi(e) = 1$ are shaded. Fig. 3 also shows a path in $\mathcal{G}^c$ such that a maximal order $\rho_{max} = \rho_u = 5$ is attained for $\phi(e) = 1$. If ϕ has double diversity then ρ_{max} will not exceed $\rho_u = 5$ for all colors $\phi(e) \in \{1, 2, \ldots, M\}$. Note that the parameters of this product code are such that $N^c/M - 4$ is also equal to 5 for a ϕ with a diversity defect.

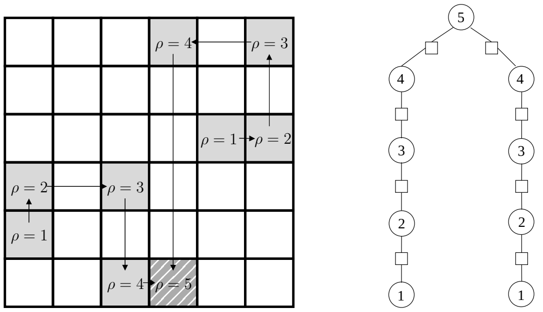


Figure 3: Compact matrix (left) and path in compact graph (right) for a product code $[12, 10]^{\otimes 2}$ showing a maximal root order of 5.

Example 2: Consider a $[14, 12] \otimes [16, 14]$ product code and a coloring ϕ with $M = 4$ colors. The compact graph has $|E^c| = 7 \times 8$ edges. The compact matrix and a path attaining $\rho = 10$ are illustrated in Fig. 4. ϕ is chosen such that the first color has a cycle involving four supersymbols. Starting from the root supersymbol ($\rho = 1$) it is possible to create a path in the graph such that $\rho = 10$ is reached. Note that a double-diversity ϕ cannot exceed a root order $\rho_u = 7$.

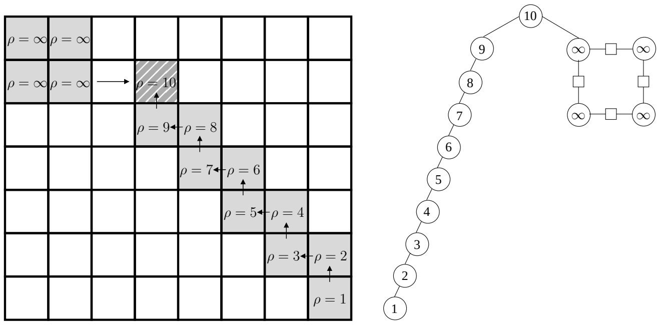


Figure 4: Compact matrix (left) and path in compact graph (right) for a product code $[14, 12] \otimes [16, 14]$ showing a maximal finite root order of 10.

The ideal situation is to construct a product code and its edge coloring in order to obtain $\rho(e) = 1$ for all edges. An analysis based on ρ_u reveals the existence of a trade-off between minimizing the number of decoding iterations and the valid range of both coding rates for the product code components. In Section IV-A, we will show unbalanced product codes where a sufficient condition on the component rates imposes order 1 to all edges. The sufficient condition is given by Lemma 1. Section IV introduces an efficient edge coloring algorithm for product codes by making use of their compact graph representation.

IV. Edge coloring algorithm under constraints

In Section III, we described graph representations of product codes and we introduced the root order $\rho(e)$ of an edge with respect to its color $\phi(e)$. Our objective is to find a coloring ϕ such that the maximum diversity order is reached under block erasures. The notion of root order in Definition 7 is for double diversity ($L = 2$) because it indirectly assumes that all symbols of one color out of M can be erased by the channel. Given the Singleton bound tradeoff stated in (7), double diversity is sufficient in distributed storage applications where the required coding rate should be sufficiently high. Definition 7 may be generalized to take into account two or more erased colors, e.g. see Figure 11 in [14] for $L = 3$ with $M = 3$ colors where an information symbol is protected by multiple root checknodes. In this paper, we restrict both Definition 7 and the design in this section to a double-diversity product code. This double diversity on a block-erasure channel is achieved if all stopping sets, as defined and counted in Section V, can be colored in a way such that at least two distinct colors are found within the symbols of a stopping set (valid for both $\mathcal{G}$ and $\mathcal{G}^c$). This task is intractable. Imagine an edge coloring ϕ designed in a way to guarantee that all weight-w stopping sets include at least two colors. This task is already very hard (or almost impossible) for a fixed w. There is no coloring design tool for non-trivial product codes to ensure that all stopping sets of all weights incorporate at least two distinct colors.

A. Hand-made edge coloring and its limitations

The aim of this section is to give more insight on designing edge coloring, before introducing the differential evolution algorithm.

The compact graph $\mathcal{G}^c$ makes the design of an edge coloring much simpler. More details will be given later in Section V-C on the relationship between graphs and stopping sets. The number of super-edges in $\mathcal{G}^c$ with the same color is N^c/M. We also know that the size, height, and width of G^c are directly related to the component and total coding rates.

Lemma 1: Let $C_P = C_1 \otimes C_2$ be a product code with a column component $C_1[n_1, k_1]_q$ and a row component $C_2[n_2, k_2]_q$ whose coding rates are $R_1 = k_1/n_1$ and $R_2 = k_2/n_2$ respectively. Assume that $n_i - k_i$ divides n_i, for $i = 1, 2$, and assume that M divides N^c. G^c admits an edge coloring ϕ such that $\rho_{max}(\phi) = 1$ if the coding rates satisfy

$$\min(R_1, R_2) \leq 1 - \frac{1}{M}. \tag{11}$$

Proof: Consider the $|V_1^c| \times |V_2^c|$ matrix representation of $\mathcal{G}^c$. A sufficient condition to get $\rho_{max}(\phi) = 1$ is to assign the N^c/M edges having the same color to a single row or a single column. The sufficient condition for $\rho_{max}(\phi) = 1$ is expressed as $N^c/M \leq \max(n_1/(n_1 - k_1), n_2/(n_2 - k_2))$, the $\max$ let us select the longest item among a row or a column. Recall also that $|V_i^c| = n_i/(n_i - k_i)$. Using (2), the sufficient condition becomes $n_1 n_2 \leq M \cdot \max(n_1(n_2 - k_2), n_2(n_1 - k_1))$. Divide by $n_1 n_2$ to get the inequality announced in the Lemma statement. ■

When the palette has $M = 4$ colors, the sufficient condition in Lemma 1 is written as $\min(R_1, R_2) \leq 3/4$. In order to achieve the block-fading Singleton bound for $M = 4$, we should take $R_1 = 3/4$ and $R_2 = 1$, i.e. the product code degenerates to a single component code. It is possible to approach $R = 3/4$ by keeping $R_1 = 3/4$ and letting $R_2 = \frac{n_2 - 1}{n_2}$ be very close to 1. In this case, the row code C_2 is a single-parity check code over F_q. The product code is very unbalanced. An example of such an unbalanced product code is

$$C_P = [12, 9, 4]_q \otimes [14, 13, 2]_q.$$

From the proof of Lemma 1, the edge coloring of $\mathcal{G}^c$ satisfying $\rho_{max} = 1$ is given by the following 4×14 matrix:

$$\begin{bmatrix} R & R & R & \ldots & R & R \\ G & G & G & \ldots & G & G \\ B & B & B & \ldots & B & B \\ Y & Y & Y & \ldots & Y & Y \end{bmatrix}, \tag{12}$$

where the colors $\phi(e) = 1, 2, 3, 4$ are replaced by the four letters 'R', 'G', 'B', and 'Y'. The rate of $[12, 9, 4]_q \otimes [14, 13, 2]_q$ is comparable to the rate of $[12, 10, 3]_q^{\otimes 2}$, $R \approx 0.69$ but it is still far from reaching three quarters as the product code $[14, 12, 3]_q \otimes [16, 14, 3]_q$. Of course, if the practical constraints allow for it, it is possible to consider an extremely unbalanced code such as the product $[12, 9, 4]_q \otimes [100, 99, 2]_q$!

Let us build balanced product codes by relaxing the constraint $\rho_{max} = 1$. We may authorize a ρ_{max} greater than 1 but not too large in order to limit the number of decoding iterations. On the other hand, the double diversity condition on the edge coloring is maintained. Firstly, let us find a hand-made edge coloring for the $[12, 10, 3]_q^{\otimes 2}$ product code with $M = 4$ colors. $\mathcal{G}^c$ has 6 left supernodes, 6 right supernodes, and a total of 36 edges. Each color is used $N^c/M = 9$ times. The hint is to place a color on the rows of the matrix representation of $\mathcal{G}^c$, row by row from the top to the bottom in a way that avoids stopping sets. The smallest stopping set is the 2×2 square. Other non-obvious stopping sets may not be visible without a tedious row-column decoding which is equivalent to determining the root order of all edges. We start with the first color 'R' and use the following number of letters per row:

$$\begin{bmatrix} R & R & R & G & B & Y \\ R & & & R & & \\ R & & & & & \\ R & & & & & \\ R & & & & & \\ R & & & & & \end{bmatrix}. \tag{13}$$

As seen above, we completed the first row with the three other colors. On the second row, we moved the second 'R' to the right to avoid a 2×2 stopping set. Next, we can start filling the second color 'G' from the third row, then the third color 'B' from the fifth row. There will be no choice for the 9 positions of 'Y'. We allow some extra permutations to avoid small stopping sets. After filling the 36 positions, we found the following hand-made edge coloring for the $[12, 10, 3]_q^{\otimes 2}$

product code:

$$\begin{bmatrix} R & R & R & G & B & Y \\ R & B & Y & R & Y & G \\ B & G & G & G & R & Y \\ Y & G & B & Y & G & R \\ R & G & B & B & B & Y \\ R & G & B & Y & Y & B \end{bmatrix}. \tag{14}$$

This coloring ϕ gives 24 super-edges of order 1 (96 edges in the non-compact graph $\mathcal{G}$) and $\rho_{max}(\phi) = 3$. Can we find a better ϕ? Yes, in Section IV-C, the DECA algorithm outputs an edge coloring with a population of 32 super-edges of order 1 (128 edges in the non-compact graph $\mathcal{G}$) and reaching $\rho_{max}(\phi) = 2$ only.

In a similar way, we attempt to build a double-diversity coloring for a well-balanced rate-$3/4$ product code, e.g. the $[14, 12, 3]_q \otimes [16, 14, 3]_q$ product code where $R_1 = 6/7$, $R_2 = 7/8$, and $R = 3/4$. The compact graph $\mathcal{G}^c$ has 7 left vertices and 8 right vertices. For $M = 4$ colors, each color is used $N^c/M = 56/4 = 14$ times. Again, we try to avoid small obvious stopping sets like 2×2, 2×3, 3×3, etc. We start by putting five 'R' on the first row, three 'R' on the second row, two 'R' on the third row, and one 'R' on the remaining rows as follows:

$$\begin{bmatrix} R & R & R & R & R & G & B & Y \\ R & & & & & R & R & \\ R & & & & & & & R \\ R & & & & & & & \\ R & & & & & & & \\ R & & & & & & & \\ R & & & & & & & \end{bmatrix}. \tag{15}$$

We repeat the same number of color entries 'G' starting on the fourth row. The color 'B' starts with five entries on the seventh row. We allow some extra permutations to avoid small stopping sets. Colors were exchanged within a row or within a column. The coloring process was tedious. Many permutations had to be applied. Some non-obvious stopping sets appeared, a computer software was used to reveal those sets (only for this task). We reached the following hand-made double-diversity edge coloring for the $[14, 12, 3]_q \otimes [16, 14, 3]_q$ product code:

$$\begin{bmatrix} Y & R & R & Y & R & G & B & R \\ R & Y & B & G & Y & R & R & B \\ B & B & B & Y & Y & R & G & R \\ R & G & G & G & G & G & B & Y \\ R & G & Y & Y & B & Y & G & G \\ G & G & R & R & B & Y & Y & Y \\ R & G & B & B & B & B & B & Y \end{bmatrix}. \tag{16}$$

This coloring gives 30 super-edges of order 1 in $\mathcal{G}^c$ (120 edges in the non-compact graph $\mathcal{G}$) and $\rho_{max}(\phi) = 5$. In Section IV-C, for the same rate-$3/4$ product code, the DECA algorithm outputs an edge coloring with a population of 40 super-edges of order 1 (160 edges in the non-compact graph $\mathcal{G}$) and reaching $\rho_{max}(\phi) = 3$ only.

B. The algorithm

We propose in this section an algorithm for product codes that searches for an edge coloring with a large number of root-order-1 edges (*good edges*) and achieving double diversity. The search is made in the ensemble of edge colorings $\Phi(E^c)$ of the compact graph $\mathcal{G}^c$. A necessary condition on the coding rate R to get double diversity is

$$R \leq 1 - \frac{1}{M}, \tag{17}$$

i.e. those satisfying inequality (7), where M is the color palette size. Codes attaining equality in (7) are referred to as MDS in the block-fading/block-erasure sense [14], [28]. The main loop of our algorithm is a differential evolution loop that mutates a fraction of the population of bad edges. The algorithm will be referred to as the Differential Edge Coloring Algorithm (DECA).

The population of bad edges is defined by the following set

$$B = \{e \in E^c : \rho(e) > 1\}. \tag{18}$$

It should be remembered that $B = B(\phi)$ because of Definition 7, but ϕ is dropped here for the sake of simplifying the notations. The number of good edges is given by

$$\eta(\phi) = |E^c \setminus B| = |\{e \in E^c : \rho(e) = 1\}|. \tag{19}$$

Among the $|B|$ bad edges, colors of a fraction of $\aleph$ edges are modified in order to maximize $\eta(\phi)$, $\aleph \in \mathbb{N}$. The fraction $\aleph/|B|$ should be large enough to allow for a population evolution but it should stay small enough in order to limit the algorithm complexity. The DECA algorithm proceeds as follows.

Initialization. The compact graph (V_1^c, V_2^c, E^c), the number of colors M, the differential evolution parameter $\aleph$, a maximum number of rounds $MaxIter$, and an initial edge coloring ϕ_0 are made ready as an input to DECA.

Pre-processing. Build all weak compositions of $\aleph$ with M parts, i.e. write $\aleph$ as the sum of M non-negative integers,

$$\aleph = \gamma_1 + \gamma_2 + \ldots + \gamma_M, \tag{20}$$

the number of weak compositions being

$$\Gamma = \binom{\aleph - M + 1}{M - 1}. \tag{21}$$

For each weak composition, prepare the Λ permutations that permute colors among the $\aleph$ edges, the total number of these permutations is

$$\Lambda(\gamma_1, \ldots, \gamma_M) = \frac{(\gamma_1 + \ldots + \gamma_M)!}{\prod_{i=1}^{M} \gamma_i!}. \tag{22}$$

This pre-processing step is completed by setting a loop counter to zero.

Differential evolution loop. This looping phase of DECA includes three main steps.

- **Edge sets initialization.** Set $\phi = \phi_0$ and $\eta_{max} = 0$. Build $B = B(\phi)$ and randomly select a subset $B_\aleph$. There is a unique weak composition $(\gamma_1, \ldots, \gamma_M)$ of $\aleph$ associated to $B_\aleph$ determined by

$$\gamma_i = |\{e \in B_\aleph : \phi(e) = i\}| . \tag{23}$$

- **Color permutations.** For $\lambda = 1 \ldots \Lambda(\gamma_1, \ldots, \gamma_M)$, replace the image of $B_\aleph$ in the mapping ϕ by a permutation of $\phi_0(B_\aleph)$. The color permutation is denoted by π^λ. This step is a modification of the mapping ϕ_0 at the $\aleph$ bad edges, i.e. $\phi(B_\aleph) \leftarrow \pi^\lambda(\phi_0(B_\aleph))$. Record the mapping with the largest number of good edges, i.e. the edge coloring with the best $\eta(\phi)$, in ϕ_1 and update η_{max}.
- **Termination.** Increment the counter of evolution loops. Stop and output ϕ_1 if this counter reaches $MaxIter$, otherwise set $\phi_0 = \phi_1$ and go back to the edge sets initialization.

A detailed functional flowchart of DECA is drawn in Figure 5. The complexity of DECA is mainly due to the differential evolution loop. The complexity is proportional to $\Lambda(\gamma_1, \ldots, \gamma_M)$ per round. Hence, the number of operations in DECA behaves as

$$\Lambda \le \Lambda_{max}(\aleph, M) = \frac{\aleph!}{((\aleph/M)!)^M}. \tag{24}$$

When $\aleph$ is not multiple of M, the denominator in the right term should be rewritten as $\prod_{i=1}^{i_0} \lfloor \aleph/M \rfloor \times \prod_{i=i_0+1}^{M} \lceil \aleph/M \rceil$, where i_0 is chosen such that the sum of all elements involved in both products is equal to $\aleph$. All Γ compositions of $\aleph$ are not considered by the algorithm. In fact, the total number of permutations for all weak compositions is

$$\sum_{j=1}^{\Gamma} \frac{(\gamma_1(j) + \ldots + \gamma_M(j))!}{\prod_{i=1}^{M} \gamma_i(j)!} = M^\aleph. \tag{25}$$

Fortunately, the per-round complexity of DECA given in (24) is much smaller that $M^\aleph$, i.e. $\Lambda_{max} = o(M^\aleph)$. In practical product code design, we will also have $\Lambda_{max} \ll M^\aleph \ll M^{N^c}$.

The proposed edge coloring algorithm aims at maximizing $\eta(\phi)$ but does not guarantee that $\forall e \in E^c, \rho(e) < \infty$. In some cases, the algorithm may terminate all its rounds with some edges having an infinite order, i.e. the coloring is not double-diversity. This occurs when trying to design a product code with a coding rate very close or equal to $1 - 1/M$, the block-fading/block-erasure Singleton bound rate. To remedy for this weakness, DECA is endowed with an extra subroutine called *Max Diversity*, as shown in Figure 5. Likewise the second step in the differential evolution loop, this subroutine applies color permutations to a subset $B_{\aleph_1}$ of edges, $|B_{\aleph_1}| = \aleph_1$, $B_{\aleph_1} \subset B^\infty$, and

$$B^\infty = \{e \in E^c : \rho(e) = \infty\}. \tag{26}$$

C. Applications

Now, let us apply DECA to design two double-diversity product codes with MDS components. Numerical values are selected to make these codes suitable to distributed storage applications and to diversity systems in wireless networks. The parameter $MaxIter$ is 100. DECA with its hundred iterations runs in a small fraction of a second on a standard computer machine.

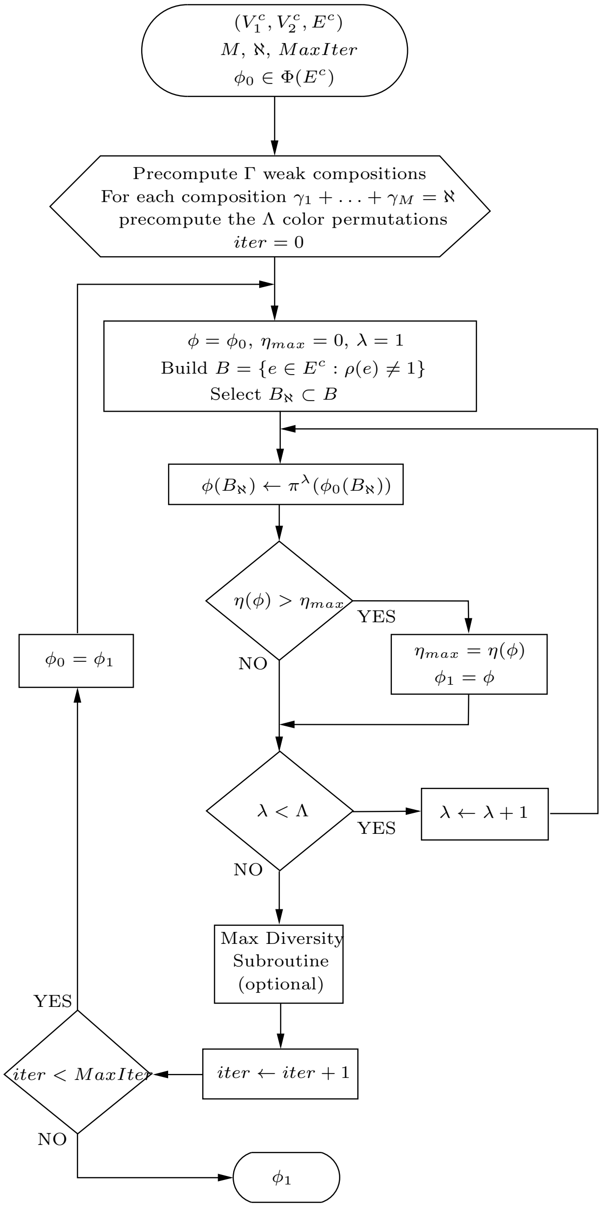
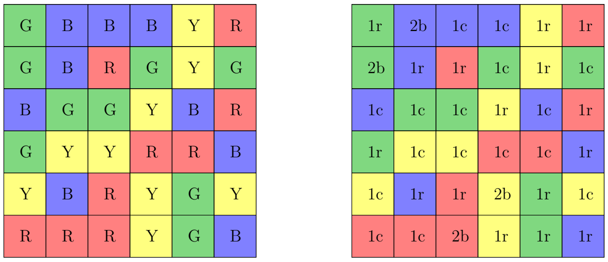

(V_1^c, V_2^c, E^c)
M, $\aleph$, $MaxIter$
$\phi_0 \in \Phi(E^c)$
Precompute Γ weak compositions
For each composition $\gamma_1 + \ldots + \gamma_M = \aleph$
precompute the Λ color permutations
$iter = 0$
$\phi = \phi_0$, $\eta_{max} = 0$, $\lambda = 1$
Build $B = \{e \in E^c : \rho(e) \neq 1\}$
Select $B_\aleph \subset B$
$\phi(B_\aleph) \leftarrow \pi^\lambda(\phi_0(B_\aleph))$
$\eta(\phi) > \eta_{max}$
YES
NO
$\eta_{max} = \eta(\phi)$
$\phi_1 = \phi$
$\phi_0 = \phi_1$
$\lambda < \Lambda$
YES
NO
$\lambda \leftarrow \lambda + 1$
Max Diversity
Subroutine
(optional)
YES
$iter < MaxIter$
$iter \leftarrow iter + 1$
NO
ϕ_1

Figure 5: Flowchart of the edge coloring algorithm (DECA) for designing double-diversity product codes.

Example 3: The first application of DECA is to color edges in the compact graph of $C_{P1} = [n, k, d]_q^{\otimes 2}$, where $n = 12$, $k = 10$, $d = 3$, and the finite-field alphabet size is $q > 12$. The coding rate of C_{P1} is $R(C_{P1}) = 25/36 < 1 - 1/M = 3/4$, i.e. the gap to (7) is 1/18. This small gap is enough to render an uncomplicated double-diversity design. The coloring in $\Phi(E^c)$ can be easily converted into its counterpart in $\Phi(E)$ by replacing each supersymbol with 4 symbols. From (5) and (6), the total number of edge colorings is $|\Phi(E)| \approx 10^{83}$ in the non-compact graph and $|\Phi(E^c)| \approx 10^{19}$ in the compact graph. The differential evolution parameter $\aleph$ is set to 8. The diversity subroutine is deactivated. We have

$$\Lambda_{max}(8,4) = 2520 \ll |\Phi(E^c)| \ll |\Phi(E)|.$$

For almost any choice of the initial coloring ϕ_0 uniformly distributed in $\Phi(E^c)$, DECA yields a double-diversity coloring ϕ_1. For roughly one choice out of three for ϕ_0, the algorithm outputs a coloring ϕ_1 such that $\eta(\phi_1) \geq 28$. Figure 6a shows the matrix representation of a special ϕ_1 found by DECA. It has $\eta(\phi_1) = 32$ which corresponds to $\eta = 128$ in (V_1, V_2, E). The corresponding rootcheck order matrix is shown in Figure 6b. The highest attained order for this coloring is $\rho_{max}(\phi) = 2$. The maximal order for all colorings in $\Phi(E^c)$ from Theorem 1 is $\rho_u = 5$. This coloring satisfies equality in (8) since $2\rho_{max}(\phi) + \eta_{min}(\phi) = 12$.



G	B	B	B	Y	R
G	B	R	G	Y	G
B	G	G	Y	B	R
G	Y	Y	R	R	B
Y	B	R	Y	G	Y
R	R	R	Y	G	B

(a)

1r	2b	1c	1c	1r	1r
2b	1r	1r	1c	1r	1c
1c	1c	1c	1r	1c	1r
1r	1c	1c	1c	1c	1r
1c	1r	1r	2b	1r	1c
1c	1c	2b	1r	1r	1r

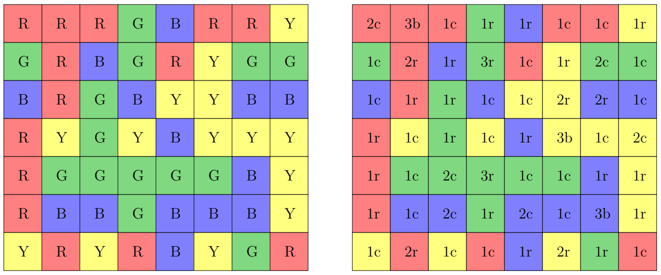
(b)

Figure 6: Compact coloring matrix (figure a) and the corresponding rootcheck-order matrix (figure b) for the $[12,10]^{\otimes 2}$ product code C_{P1} found by DECA, $\eta(\phi) = 32$ and $\rho_{max} = 2$.

Example 4: The second more challenging application of DECA is the design of a double-diversity product code attaining the block-fading/block-erasure Singleton bound. Let us consider $C_{P2} = [n_1, k_1, d_1]_q \otimes [n_2, k_2, d_2]_q$, where $n_1 = 14$, $k_1 = 12$, $n_2 = 16$, $k_2 = 14$, $d_1 = d_2 = 3$, and the finite-field alphabet size is $q > 16$. The coding rate is $R(C_{P2}) = 1 - 1/M = 3/4$. From (5) and (6), the total number of edge colorings is $|\Phi(E)| \approx 10^{131}$ in the non-compact graph and $|\Phi(E^c)| \approx 10^{31}$ in the compact graph. The differential evolution parameter $\aleph$ is set to 7. The diversity subroutine is activated with $\aleph_1 = 8$. We have

$$\Lambda_{max}(7,4) + \Lambda_{max}(8,4) = 3150 \ll |\Phi(E^c)| \ll |\Phi(E)|.$$

The initial coloring ϕ_0 is taken to be uniformly distributed in $\Phi(E^c)$. For almost three ϕ_0 choices out of four, DECA yields a double-diversity coloring ϕ_1. Roughly one ϕ_0 choice out of two guarantees $\eta(\phi_1) \geq 34$. Figure 7a shows the matrix representation of a special ϕ_1 found by DECA. It has $\eta(\phi_1) = 40$ which corresponds to $\eta = 160$ in (V_1, V_2, E). The rootcheck order matrix is shown in Figure 7b. The highest attained order for this coloring is $\rho_{max}(\phi) = 3$. The maximal order for all colorings in $\Phi(E^c)$ from Theorem 1 is $\rho_u = 7$. This coloring satisfies $2\rho_{max}(\phi) + \eta_{min}(\phi) = 16$ while the right term in (8) is 17.



R	R	R	G	B	R	R	Y
G	R	B	G	R	Y	G	G
B	R	G	B	Y	Y	B	B
R	Y	G	Y	B	Y	Y	Y
R	G	G	G	G	G	B	Y
R	B	B	G	B	B	B	Y
Y	R	Y	R	B	Y	G	R

(a)

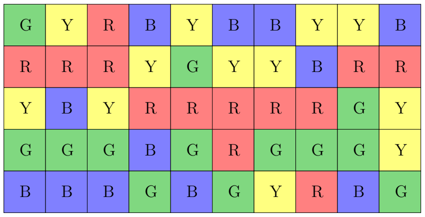
2c	3b	1c	1r	1r	1c	1c	1r
1c	2r	1r	3r	1c	1r	2c	1c
1c	1r	1r	1c	1c	2r	2r	1c
1r	1c	1r	1c	1r	3b	1c	2c
1r	1c	2c	3r	1c	1c	1r	1r
1r	1c	2c	1r	2c	1c	3b	1r
1c	2r	1c	1c	1r	2r	1r	1c

(b)

Figure 7: Compact coloring matrix (figure a) and the corresponding rootcheck-order matrix (figure b) for the $[14,12] \otimes [16,14]$ product code C_{P2} found by DECA, $\eta(\phi) = 40$ and $\rho_{max} = 3$.

Example 5: A third example suitable for nowadays distributed storage warehouses is $C_{P3} = [10,8,3]_q \otimes [10,9,2]_q$. The coding rate is $R = 18/25 = 0.72$ with a minimum distance $d_1 d_2 = 6$ and the locality is $n_1 - 1 = n_2 - 1 = 9$, i.e. this code is an improvement to the standard $RS[14,10]$ used by Facebook [49]. However, from locality point of view only, better codes are found under sequential recovery [46] as given by the bound and the construction proposed in [5] for multiple erasures. A locality of 9 is achieved for a higher rate of 0.8010 and a better locality of 6 is possible at an equivalent rate of 0.7176, see Theorem 2 in [5]. The drawback of such locally repairable codes is the loss of diversity and hence the loss of block-erasure filling capacity attained by our edge-colored product codes.

The coloring ensembles have sizes $|\Phi(E)| \approx 10^{57}$ and $|\Phi(E^c)| \approx 10^{27}$ respectively. The DECA algorithm produced double-diversity edge colorings where we distinguish two classes: a first class of colorings with $\rho_{max} = 3$ and $\eta(\phi) = 41$, and a second class with $\rho_{max} = 2$ and $\eta(\phi) = 40$. An edge coloring of the second class is shown in Figure 8. The reader is invited to determine the rootcheck order matrix and verify that 40 super-edges have root order 1 and 10 super-edges have a root order equal to 2.



G	Y	R	B	Y	B	B	Y	Y	B
R	R	R	Y	G	Y	Y	B	R	R
Y	B	Y	R	R	R	R	R	G	Y
G	G	G	B	G	R	G	G	G	Y
B	B	B	G	B	G	Y	R	B	G

Figure 8: Compact coloring matrix for the $[10,8] \otimes [10,9]$ product code found by DECA, $\eta(\phi) = 40$ and $\rho_{max} = 2$.

In figures of the previous examples, the four colors were also indicated by the first letter of the color name, Red, Green, Blue, and Yellow. The rootcheck order $\rho(e)$ for an edge e in E^c (which is also the order of the four code symbols associated to that edge) is indicated by an integer in the right part of each figure for the first two examples. In the rootcheck order matrix, $2r$ means that this supersymbol has order 2 and its root checknode is a row. Similarly, $2c$ designates a supersymbol with order 2 and a column rootcheck. The letter 'b' is written when a supersymbol has both rootchecks, a row and a column rootcheck.

D. Random edge coloring

The efficiency of the DECA algorithm was validated in the previous section in terms of number of edges of first order and the maximal order over all edges. Clearly, while evolving from one coloring to another in order to get a large $\eta(\phi)$, DECA also produced a very small maximal order $\rho_{max}(\phi)$. Any deterministic construction seems to be destined to fail given the huge size of the ensembles $\Phi(E)$ and $\Phi(E^c)$.

In this sub-section, another way to show the efficiency of our coloring algorithm is to make random selections from $\Phi(E)$ and $\Phi(E^c)$ and get an estimate of the probability distributions of $\eta(\phi)$ and $\rho_{max}(\phi)$. Indeed, a uniformly distributed permutation in the symmetric group of order N yields a uniformly distributed edge coloring ϕ in $\Phi(E)$. This is also true for $\Phi(E^c)$ when the symmetric group has order N^c. Thus, in a uniform manner, we selected 2 billion edge colorings through our computer application from $\Phi(E)$ and $\Phi(E^c)$ respectively. For each coloring, rootcheck orders of all edges were computed, i.e. for the N edges in the non-compact graph and the N^c edges in the compact graph. Only double-diversity colorings are counted in this comparison, i.e. colorings with at least one edge of infinite rootcheck order are excluded. As an illustration, the characteristics of double-diversity random coloring for C_{P1} are plotted in Figure 9 where numerical estimations of all probability distributions are compared to colorings designed via DECA.

Double diversity design is more arduous for the rate-$3/4$ C_{P2} product code than for the rate-$25/36$ C_{P1} product code because of the rate-diversity tradeoff given by the Singleton bound. For C_{P1}, the $[12,10]^{\otimes 2}$ code, 8.97% of uniformly sampled colorings have double diversity in $\Phi(E^c)$, whereas this fraction is 43.6% in $\Phi(E)$. For C_{P2}, the $[14,12]\otimes[16,14]$ code, only 0.00039% of uniformly sampled colorings have double diversity in $\Phi(E^c)$, and we found no double-diversity colorings in $\Phi(E)$ despite the 2 billion samples. As expected, compact graphs exhibit better characteristics than non-compact graphs thanks to their simpler structure, i.e. $n_i - k_i$ parity symbols are grouped inside a unique supersymbol: for C_{P1}, one double-diversity random coloring has $\eta(\phi) = 88$, $\rho_{max}(\phi) = 4$ for non-compact graphs, seven double-diversity colorings have $\eta(\phi) = 120$, and $\rho_{max}(\phi) = 2$ for compact graphs. There exists a double-diversity coloring in $\Phi(E)$ with $\rho_{max}(\phi) = 3$ but its η is 85. The estimated probability mass functions for C_{P1} are plotted in Figures 9a and 9b. For C_{P2}, one double-diversity random coloring reached $\eta(\phi) = 128$ and $\rho_{max}(\phi) = 4$ out of the 2 billion samples from $\Phi(E^c)$. In all cases, for both η and ρ, double-diversity random colorings are not as efficient as colorings designed via the DECA algorithm. The situation is worse for random colorings if a double-diversity code with maximal rate $1 - 1/M$ is to be designed. The DECA algorithm exhibits excellent values, $\eta = 160$ and $\rho = 3$, for the rate-$3/4$ $[14,12]\otimes[16,14]$ product code.

In the next section we analyze stopping sets in product codes with MDS components, we describe the relationship between stopping sets and the product code graph representation, and finally we enumerate obvious and non-obvious stopping sets. Stopping sets enumeration is useful to determine the performance of a product code with and without edge coloring.

V. Stopping sets for MDS components

The purpose of this section is to prepare the way for determining the performance of iterative decoding of non-binary product codes. The analysis of stopping sets in a product code will yield a tight upper bound of its iterative decoding performance over a channel with independent erasures. The same analysis will be useful to accurately estimate the performance under edge coloring in presence of block and multiple erasure channels.

A. Decoding erasures

Definition 8: An erasure pattern is said to be ML-correctable if the ML decoder is capable of solving all its erased symbols.

For an erasure pattern which is not correctable under ML or iterative decoding, the decoding process may fill none or some of the erasures and then stay stuck on the remaining ones. Before describing the stopping sets of a product code, let us recall some fundamental results regarding the decoding of its row and column component codes. The ML erasure-filling capability of a linear code satisfies the following property.

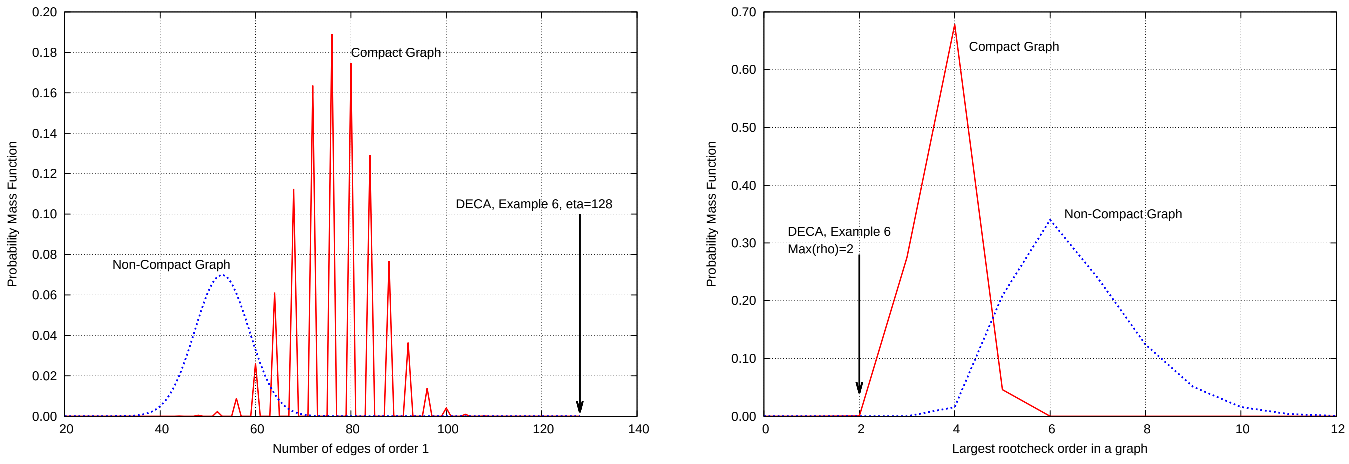
Proposition 1: Let $C[n,k,d]_q$ be a linear code with $q \geq 2$. Assume that C is not MDS and the n symbols of a codeword are transmitted on an erasure channel. Then, there exists an erasure pattern of weight greater than $d-1$ that is ML-correctable.

Proof: Let H be an $(n-k)\times n$ parity-check matrix of C. We have $n-k > d-1$ because C is not MDS. For any integer w in the range $[d, n-k]$, there exists a set of w linearly independent columns in H. Choose an erasure pattern of weight w with erasures located at the positions of the w independent columns. Then, the ML decoder is capable of solving all these erasures by simple Gaussian reduction of H. ■

For MDS codes, based on a proof similar to the proof of Proposition 1, we state a well-known result in the following corollary.

Corollary 2: Let $C[n,k,d]_q$ be an MDS code. All erasure patterns of weight greater than $d-1$ are not ML-correctable.

We conclude from the previous corollary that an algebraic decoder for an MDS code attains the word-error performance of its ML decoder. What about symbol-error performance? Indeed, for general binary and non-binary codes, the ML decoder may outperform an algebraic decoder since it is capable of filling some of the erasures when dealing with a pattern which is not ML-correctable. In the MDS case, the answer comes from the absence of spectral holes for any MDS



(a)

(b)

Figure 9: Distribution of $\eta(\phi)$ (figure a) and $\rho_{max}(\phi)$ (figure b) for double-diversity random edge colorings uniformly distributed in $\Phi(E)$ and $\Phi(E^c)$. Product code $[12, 10]^{\otimes 2}$.

code beyond its minimum distance. This basic result is proven via standard tools from algebraic coding theory [8], [41]:

Proposition 2: Let $C[n, k, d]_q$ be a non-binary MDS code ($q > n > 2$). For any w satisfying $d \leq w \leq n$ and any support $\mathcal{X} = \{i_1, i_2, \ldots, i_w\}$, where $1 \leq i_j \leq n$, there exists a codeword in C of weight w having $\mathcal{X}$ as its own support.

Proof: By assumption we have $w > r = n - k$. Let H be a parity-check matrix of C with rank $r = n - k$. Recall that the MDS property makes full-rank any set of $n - k$ columns of H [41]. w is written as $w = r + \ell$, where $\ell = 1 \ldots k$. The w positions of $\mathcal{X}$ are anywhere inside the range $[1, n]$, but for simplicity let us denote $h_1 \ldots h_r$ the r columns of H in the first r positions. The last ℓ columns are denoted $\zeta_1 \ldots \zeta_\ell$. For any $j = 1 \ldots \ell$, we have

$$\zeta_j = \sum_{i=1}^{r} a_{i,j} h_i,$$

where $a_{i,j} \in \mathbb{F}_q \setminus \{0\}$ otherwise it contradicts $d = n - k + 1$. Now, select $\alpha_1 \ldots \alpha_\ell$ from $\mathbb{F}_q \setminus \{0\}$ such that: α_1 is arbitrary, α_2 is chosen outside the set $\{-\alpha_1 a_{i,1}/a_{i,2}\}_{i=1}^{r}$, then α_3 is chosen outside the set $\{(-\alpha_1 a_{i,1} - \alpha_2 a_{i,2})/a_{i,3}\}_{i=1}^{r}$, and so on, up to α_ℓ which is chosen outside the set $\{-\sum_{u=1}^{\ell-1} \alpha_u a_{i,u}/a_{i,\ell}\}_{i=1}^{r}$. Here, the notation a/b in $\mathbb{F}_q \setminus \{0\}$ is equivalent to the standard algebraic notation ab^{-1}. The equality

$$\sum_{j=1}^{\ell} \alpha_j \zeta_j = \sum_{i=1}^{r} \sum_{j=1}^{\ell} \alpha_j a_{i,j} h_i$$

produces a codeword of Hamming weight w. Hence, there exists a codeword of weight w with non-zero symbols in all positions given by $\mathcal{X}$. ■

Now, at the symbol level for an MDS code and an erasure pattern which is not ML-correctable ($w > d-1$), we conclude from Proposition 2 that the ML decoder cannot solve any of the w erasures because they are covered by a codeword. Consequently, an algebraic decoder for an MDS code also attains the symbol-error performance of the ML decoder. This behavior will have a direct consequence on the iterative decoding of a product code with MDS components: stopping sets are identical when dealing with algebraic and ML-per-component decoders.

A general description of a stopping set was given by Definition 1. The exact definition of a stopping set depends on the iterative decoding type. For product codes, four decoding methods are known:

- Type I: ML decoder. This is a non-iterative decoder. It is based on a Gaussian reduction of the parity-check matrix of the product code.
- Type II: Iterative algebraic decoder. At odd decoding iterations, component codes C_1 on each column are decoded via an algebraic decoder (bounded-distance) that fills up to $d - 1$ erasures. Similarly, at even decoding iterations, component codes C_2 on each row are decoded via an algebraic decoder.
- Type III: Iterative ML-per-component decoder. This decoder was considered by Rosnes in [53] for binary product codes. At odd decoding iterations, column codes C_1 are decoded via an optimal decoder (ML for C_1). At even decoding iterations, row codes C_2 are decoded via a similar optimal decoder (ML for C_2).
- Type IV: Iterative belief-propagation decoder based on the Tanner graph of $C_{\mathcal{P}}$, as studied by Schwartz et al. for general linear block codes [56] and by Di et al. for low-density parity-check codes [18].

The three iterative decoders listed above give rise to three different kinds of stopping sets. As previously indicated, from Corollary 2 and Propositions 2, we concluded that type-II and type-III stopping sets are identical if component codes are MDS.

B. *Stopping set definition*

Let C be a q-ary linear code of length n, i.e. C is a sub-space of dimension k of $\mathbb{F}_q^n$. The support of C, denoted by $\mathcal{X}(C)$, is the set of ℓ distinct positions

$\{i_1, i_2, \ldots, i_\ell\} = \{i_j\}_{j=1}^{\ell}$, $1 \leq i_j \leq n$, such that, for all j, there exists a codeword $c = (c_1 \ldots c_n) \in C$ with $c_{i_j} \neq 0$. This notion of support $\mathcal{X}$ is applied to rows and columns in a product code.

Now, we define a rectangular support which is useful to represent a stopping set in a bi-dimensional product code. Let $\mathcal{S} \subseteq \{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$ be a set of symbol positions in the product code. The set of row positions associated to $\mathcal{S}$ is $\mathcal{R}_1(\mathcal{S}) = \{i_1, \ldots, i_{\ell_1}\}$ where $|\mathcal{R}_1(\mathcal{S})| = \ell_1$ and for all $i \in \mathcal{R}_1(\mathcal{S})$ there exists $(i, \ell) \in \mathcal{S}$. The set of column positions associated to $\mathcal{S}$ is $\mathcal{R}_2(\mathcal{S}) = \{j_1, \ldots, j_{\ell_2}\}$ where $|\mathcal{R}_2(\mathcal{S})| = \ell_2$ and for all $j \in \mathcal{R}_2(\mathcal{S})$ there exists $(\ell, j) \in \mathcal{S}$. The rectangular support of $\mathcal{S}$ is

$$\mathcal{R}(\mathcal{S}) = \mathcal{R}_1(\mathcal{S}) \times \mathcal{R}_2(\mathcal{S}), \tag{27}$$

i.e. the smallest $\ell_1 \times \ell_2$ rectangle including all columns and all rows of $\mathcal{S}$.

Definition 9: [53] Consider a product code $C_P = C_1 \otimes C_2$. Let $\mathcal{S} \subseteq \{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$ with $|\mathcal{R}_1(\mathcal{S})| = \ell_1$ and $|\mathcal{R}_2(\mathcal{S})| = \ell_2$. Consider the ℓ_1 rows of $\mathcal{S}$ given by $\mathcal{S}_r^{(i)} = \{j : (i, j) \in \mathcal{S}\}$ and the ℓ_2 columns of $\mathcal{S}$ given by $\mathcal{S}_c^{(j)} = \{i : (i, j) \in \mathcal{S}\}$. The set $\mathcal{S}$ is a stopping set of type III for C_P if there exist linear subcodes $C_c^{(j)} \subseteq C_1$ and $C_r^{(i)} \subseteq C_2$ such that $\mathcal{X}(C_c^{(j)}) = \mathcal{S}_c^{(j)}$ and $\mathcal{X}(C_r^{(i)}) = \mathcal{S}_r^{(i)}$ for all $i \in \mathcal{R}_1(\mathcal{S})$ and for all $j \in \mathcal{R}_2(\mathcal{S})$.

The cardinality $|\mathcal{S}|$ is called the size of the stopping set and will also be referred to in the sequel as the weight of $\mathcal{S}$. Recall that type II and type III stopping sets are identical when both C_1 and C_2 are MDS. Stopping sets of type III were studied for binary product codes by Rosnes [53]. His analysis is based on the generalized Hamming distance [30], [69] because sub-codes involved in Definition 9 may have a dimension greater than 1. In the non-binary MDS case, according to Proposition 2, all these sub-codes have dimension 1, i.e. they are generated by a single non-zero codeword. Consequently, the generalized Hamming distance is not relevant when using MDS components. In such a case, the analysis of type II stopping sets is mainly combinatorial and does not require algebraic tools.

Stopping sets for decoder types II-IV can be characterized by four main properties summarized as follows.

- Obvious or not obvious sets, also known as rank-1 sets. A stopping set $\mathcal{S}$ is obvious if $\mathcal{S} = \mathcal{R}(\mathcal{S})$.
- Primitive or non-primitive stopping sets. A stopping set is primitive if it cannot be partitioned into two or more smaller stopping sets. Notice that all stopping sets, whether they are primitive or not, are involved in the code performance.
- Codeword or non-codeword. A stopping set $\mathcal{S}$ is said to be a codeword stopping set if there exists a codeword c in C_P such that $\mathcal{X}(c) = \mathcal{S}$.
- ML-correctable or non-ML-correctable. A stopping set $\mathcal{S}$ cannot be corrected via ML decoding if it includes the support of a non-zero codeword.

In the remaining material of this paper, we restrict our study to type II stopping sets.

Example 6: Consider a $[n_1, n_1 - 2, 3]_q \otimes [n_2, n_2 - 2, 3]_q$ product code. A stopping set $\mathcal{S}$ of size $w = 9$ is shown as a weight-9 matrix of size $n_1 \times n_2$, where 1 corresponds to an erased position:

$$\mathcal{S} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{28}$$

We took $n_1 = n_2 = 7$ for illustration. The rectangular support is shown in a compact representation as a matrix of size $\ell_1 \times \ell_2 = 3 \times 3$,

$$\mathcal{R}(\mathcal{S}) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{29}$$

The stopping set in (28) is obvious, it has the same size as its rectangular support. It corresponds to a matrix of rank 1. Each row and each column of $\mathcal{S}$ has weight 3. Iterative row-column decoding based on component algebraic decoders fails in decoding rows and columns since the number of erasures exceeds the erasure-filling capacity of the MDS components. This stopping set is not ML-correctable because it is a product-code codeword. In the sequel, all stopping sets (type II) shall be represented in this compact manner by a smaller rectangle of size $\ell_1 \times \ell_2$.

Example 7: For the same $[n_1, n_1 - 2, 3]_q \otimes [n_2, n_2 - 2, 3]_q$ product code used in the previous example, the following stopping sets of size 12 are not obvious.

$$\mathcal{S}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{30}$$

$$\mathcal{S}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{31}$$

In compact form, their rectangular support is

$$\mathcal{R}(\mathcal{S}_1) = \mathcal{R}(\mathcal{S}_2) = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}. \tag{32}$$

These stopping sets have size 12 and a 4×4 rectangular support. For $w = 12$, it is also possible to build an obvious stopping set in a 3×4 rectangle or a 4×3 rectangle full of 1. $\mathcal{S}_1$ is ML-correctable since it does not cover a product code codeword. $\mathcal{S}_2$ covers a codeword hence it is not ML-correctable.

C. Stopping sets and subgraphs of product codes

A stopping set as defined by Definition 9 corresponds to erased edges in the non-compact graph $\mathcal{G}$ introduced in Section III-A. Indeed, consider the size-9 stopping set given by (28) or (29). The nine symbol positions involve nine edges in $\mathcal{G}$, three row checknodes, and three column checknodes. Each of these six checknodes has three erased symbols making the $[12, 10, 3]$ decoder fail. This stopping set is equivalent to a subgraph of 9 edges in $\mathcal{G}$ as shown in Figure 10.

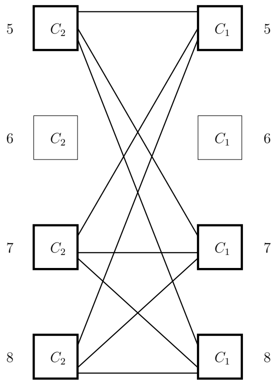


Figure 10: A sub-graph of $\mathcal{G}$ representing the size-9 obvious stopping set. The graph $\mathcal{G}$ has $|E| = 144$ edges, $|V_2| = 12$ left (row) checknodes, and $|V_1| = 12$ left (column) checknodes. Only the stopping set edges are drawn.

The subgraph in Figure 10 has three length-4 cycles and two length-6 cycles. The small cycles of length-4 are associated to an erasure pattern with a 2×2 rectangular support which is not a stopping set ($d_1 = d_2 = 3$). Similarly, length-6 cycles are not stopping sets and are associated to erasure patterns with a 2×3 rectangular support. We will see in the next section that the minimum stopping set size is $d_1 d_2 = 9$, i.e. it is equal to the minimum Hamming distance of the product code.

A subgraph of $\mathcal{G}^c$ can be embedded into $\mathcal{G}$ by splitting each super-edge into $(n_1 - k_1) \times (n_2 - k_1)$ edges. The converse is not always true. The subgraph with nine edges in Figure 10 cannot be compressed into a subgraph of $\mathcal{G}^c$. For the $[12, 10, 3]^{\otimes 2}$ product code, a supersymbol in $\mathcal{G}^c$ contains four edges. Hence, a necessary condition for a stopping set in $\mathcal{G}$ to become a valid stopping set in $\mathcal{G}^c$ is to erase edges in groups of 4. Knowing that type II and type III stopping sets are identical when row and column codes C_1 and C_2 are MDS, Definition 9 leads to the following corollaries.

Corollary 3: Let $C_P = C_1 \otimes C_2$ be a product code with MDS components C_1 and C_2 having minimum Hamming distance d_1 and d_2 respectively. Assume that symbols (edges) of $\mathcal{G} = (V_1, V_2, E)$ are sent over an erasure channel. A stopping set for the iterative decoder is a subgraph of $\mathcal{G}$ such that all column vertices in V_1 have a degree greater than or equal to d_1 and all row vertices in V_2 have a degree greater than or equal to d_2.

Corollary 4: Let $C_P = C_1 \otimes C_2$ be a product code with MDS components C_1 and C_2 having minimum Hamming distance d_1 and d_2 respectively. Assume that supersymbols (super-edges) of $\mathcal{G}^c = (V_1^c, V_2^c, E^c)$ are sent over an erasure channel. A stopping set for the iterative decoder is a subgraph of $\mathcal{G}^c$ such that all column vertices in V_1^c have a degree greater than or equal to 2 and all row vertices in V_2^c have a degree greater than or equal to 2.

The above corollaries suppose a symbol (or a supersymbol) channel with independent erasures. When $\mathcal{G}$ is endowed with an edge coloring ϕ, we get the same constraint on the validity of a subgraph embedding from $\mathcal{G}^c$ into $\mathcal{G}$. We know from Section III-A that $\Phi(E^c \to E)$ is a subset of $\Phi(E)$, i.e. some edge colorings of $\mathcal{G}$ are not edge colorings of $\mathcal{G}^c$. Consequently, on a block-erasure channel, if all super-edges of the same color are erased, stopping sets in $\mathcal{G}^c$ are a subset of those in $\mathcal{G}$. The non-compact graph $\mathcal{G}$ has a larger ensemble of stopping sets, with or without edge coloring. As an example, for the $[12, 10, 3]^{\otimes 2}$ product code, the smallest stopping set in $\mathcal{G}^c$ has size 2×2 when four super-edges are erased which yields a stopping set of size 16 in $\mathcal{G}$.

Example 8: Consider the $[9, 6, 4]_q^{\otimes 2}$ product code where $d_1 = d_2 = 4$ and $q > 9$. Assume that our palette has $M = 3$ colors. The non-compact graph admits an ensemble of $|\Phi(E)| = 4490186382903298862950669893074864640$ edge colorings! The compact graph has $|\Phi(E^c)| = 1680$ only. In $\mathcal{G}^c$, each color is used $N^c/M = 3$ times. For a channel erasing all symbols of the same color, the compact graph has no stopping sets (the 2×2 rectangular support cannot be filled by a single color). A compact matrix representation of $\mathcal{G}^c$ attaining double diversity with all symbols of order 1 is given by the trivial matrix

$$\begin{bmatrix} R & G & B \\ B & R & G \\ G & B & R \end{bmatrix}, \tag{33}$$

where the color $\phi(e) = 1$ is replaced by the letter 'R', $\phi(e) = 2$ is replaced by the letter 'G', and $\phi(e) = 3$ is replaced by the letter 'B'. The non-compact graph has 9×9 edges, each color is used 27 times. Double diversity is lost in $\mathcal{G}$ if one of the 4×4, 4×5, or 5×5 obvious stopping sets is covered by a unique color. Clearly, $\mathcal{G}^c$ makes the design much easier. This double-diversity product code has a relatively low coding rate. More challenging product code designs are given in Section IV with higher rates up to the one imposed by the block-fading/block-erasure Singleton bound.

D. Enumeration of stopping sets

For a fixed non-zero integer w, the number of stopping sets of size w, denoted as τ_w, falls in two different cases. Firstly, $\tau_w = 0$ if w is small with respect to the minimum Hamming distance of the product code. Also, $\tau_w = 0$ for special erasure patterns obtained by adding a small neighborhood to a smaller obvious set. Secondly, for both obvious and non-obvious stopping sets, τ_w is non-zero and the weight w may correspond to many rectangular supports of different height and width. The code performance over erasure channels is dominated by not-so-large stopping sets. Non-empty stopping sets of the second case satisfy the general property stated in the following lemma.

Lemma 2: Given a weight $w \leq (d_1+1)(d_2+1)$ and assuming $\tau_w > 0$, then $\exists \mathcal{S}^0$ such that $\forall \mathcal{S}$ with $|\mathcal{S}| = w$, we have $\|\mathcal{R}(\mathcal{S})\| \leq \|\mathcal{R}(\mathcal{S}^0)\| = (\ell_1^0, \ell_2^0)$, where

$$\ell_1^0 \leq d_1 + 1 + \left\lfloor \frac{d_1+1}{d_2} \right\rfloor, \tag{34}$$

$$\ell_2^0 \leq d_2 + 1 + \left\lfloor \frac{d_2+1}{d_1} \right\rfloor. \tag{35}$$

Proof: Let w be equal to $(d_1+1)(d_2+1)$. In order to establish an upper bound of the height ℓ_1, we build the highest possible rectangular support for this weight w. Assume the rectangle is $\ell_1^0 \times \ell_2$, each of its rows should have at least d_2 erasures to make the type-II decoder fail. Then $d_2\ell_1^0 \leq (d_1+1)(d_2+1)$ which becomes the upper bound given by (34). Now, if w is less than $(d_1+1)(d_2+1)$, the rectangular support of the stopping set can only shrink in size. The upper bound of the width in (35) is proven in a similar way. ■

The above lemma states the existence of a maximal rectangular support for a given stopping set size. The example given below cites stopping sets with a unique-size rectangular support and stopping sets with multiple-size rectangular supports.

Example 9: Consider a $C_1 \otimes C_2$ product code where C_1 and C_2 are both MDS with minimum Hamming distance 3. The stopping set given by (29) cannot have a large rectangular support. In general, all stopping sets of size $d_1 d_2$ have a rectangular support of fixed dimensions $d_1 \times d_2$. Now, let $w = 12$. As indicated in Example 7, stopping sets of size 12 may be included in rectangular supports of dimensions 3×4, 4×3, and 4×4. For $w = 12$, it is impossible to build a 4×5 rectangular support (reductio ad absurdum) making $\ell_1^0 = 4$ and $\ell_2^0 = 4$. A similar proof by contradiction yields $\ell_1^0 = 5$ and $\ell_2^0 = 5$ for $w = 15$.

The next lemma gives an obvious upper bound of the size of $\mathcal{R}(\mathcal{S})$ by stating a simple limit on the number of zeros (non-erased positions) inside $\mathcal{R}(\mathcal{S})$.

Lemma 3: Let $\mathcal{R}(\mathcal{S})$ be the $\ell_1 \times \ell_2$ rectangular support of a stopping set $\mathcal{S}$ of size w. Let $\beta = \ell_1\ell_2 - w$ be the number of zero positions, or equivalently β is the size of the set $\mathcal{R}(\mathcal{S}) \backslash \mathcal{S}$. Then

$$\beta \leq \min((\ell_1 - d_1)\ell_2, \ell_1(\ell_2 - d_2)). \tag{36}$$

Before stating and proving Theorem 2, we announce two results in Lemma 4 and Lemma 5 on bipartite graphs enumeration. We saw in the previous section that stopping sets are sub-graphs of $\mathcal{G}$ and $\mathcal{G}^c$, see Corollary 3 and Corollary 4. In other words, the enumeration of stopping sets represented as matrices of a given distribution of row weight and column weight is equivalent to enumerating bipartite graphs where left vertices stand for rows and right vertices stand for columns. An edge should be drawn between a left vertex and a right vertex according to some rule, e.g. the rule used in the previous section draws an edge in the bipartite graph for each 1 in the stopping set matrix. Stopping sets enumeration in the next theorem is based on β, the number of zeros or the number of non-erased positions. Hence, we shall use the opposite rule. A stopping set of weight w and having a $\ell_1 \times \ell_2$ rectangular support shall be represented by a bipartite graph with ℓ_1 left vertices, ℓ_2 right vertices, and a total of $\beta = \ell_1\ell_2 - w$ edges. Notice that these bipartite graphs have no length-2 cycles because parallel edges are forbidden.

For finite ℓ_1 and ℓ_2, given the left degree distribution and the right degree distribution, there exists no exact formula for counting bipartite graphs. The best recent results are asymptotic in the graph size for sparse and dense matrices [15], [17] and cannot be applied in our enumeration. The following two lemmas solve two cases encountered in Theorem 2 for $w = d(d+2)$ and $w = (d+1)(d+1)$ both inside a $(d+2) \times (d+2)$ rectangular support. The definition of special partitions is required before introducing the two lemmas.

Definition 10: Let $\ell \geq 2$ be an integer. A *special partition* of length j of ℓ is a partition defined by a tuple $(\ell_1, \ell_2, \ldots, \ell_j)$ such that its integer components satisfy:

- $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_j$.
- $\sum_{i=1}^{j} \ell_i = \ell$.
- $\ell_i \geq 2, \forall j$.
- $1 \leq j \leq \ell/2$.

A special partition shall be denoted by $((\ell_1, \ldots, \ell_j))$.

Definition 11: The *group number* of a special partition, denoted by $\kappa = \kappa(\ell_1, \ell_2, \ldots, \ell_j)$, is the number of different integers ℓ_j, for $j = 1 \ldots \ell/2$. In other words, following set theory, the set including the j integers ℓ_i's is $\{\ell_{i_1}, \ell_{i_2}, \ldots, \ell_{i_\kappa}\}$. The group number divides the partition of ℓ into κ groups where the m^{th} group includes ℓ_{i_m} repeated g_m times, and $\sum_{m=1}^{\kappa} g_m = j$.

Lemma 4: Consider bipartite graphs defined as follows: ℓ left vertices, ℓ right vertices, all vertices have degree 2, and no length-2 cycles are allowed. For $\ell \geq 2$, the total number x_ℓ of such bipartite graphs is given by the expression

$$x_\ell = \sum_{((\ell_1,\ldots,\ell_j))} \frac{1}{\prod_{m=1}^{\kappa(\ell_1,\ldots,\ell_j)} g_m!} \prod_{k=1}^{j} \frac{\prod_{u=0}^{\ell_k-1}(\ell - \sum_{i=1}^{k-1}\ell_i - u)^2}{2\ell_k} \tag{37}$$

where $\sum_{((\ell_1,\ldots,\ell_j))}$ is a summation over all special partitions of the integer ℓ, $\kappa(\ell_1, \ldots, \ell_j)$ is the group number of the special partition $((\ell_1, \ldots, \ell_j))$, and g_m is the size of the m^{th} group.

Proof: Firstly, let us find the number of Hamiltonian bipartite graphs having ℓ_k left vertices, ℓ_k right vertices, all vertices of degree 2, and no length-2 cycles allowed. There are $(\ell_k!)^2$ ways to choose the order of all left and right vertices. If the Hamiltonian cycle is represented by a sequence of $2\ell_k$ integers corresponding to the $2\ell_k$ vertices of the bipartite graph, then there are $2\ell_k$ ways to shift the Hamiltonian cycle without changing the graph. Hence, the number of Hamiltonian bipartite graphs of degree 2 is

$$\frac{(\ell_k!)^2}{2\ell_k}. \tag{38}$$

Secondly, given the half-size ℓ of the bipartite graph stated in this lemma, all special partitions of ℓ are considered. For a fixed special partition $((\ell_1, \ell_2, \ldots, \ell_j))$ the bipartite graph is decomposed into j Hamiltonian graphs each of length ℓ_k,

$k = 1 \dots j$. The number of choices for selecting the vertices of the j Hamiltonian graphs is

$$\prod_{k=1}^{j} \binom{\ell - \sum_{i=1}^{k-1} \ell_i}{\ell_k}^2. \tag{39}$$

The above number should be multiplied by the number of Hamiltonian graphs for each selection of vertices to get

$$\prod_{k=1}^{j} \binom{\ell - \sum_{i=1}^{k-1} \ell_i}{\ell_k}^2 \frac{(\ell_k!)^2}{2\ell_k}. \tag{40}$$

But for a given special partition, each group of size g_m is creating $g_m!$ identical bipartite graphs. Hence, the final result for a fixed partition becomes

$$\frac{1}{\prod_{m=1}^{\kappa(\ell_1,\dots,\ell_j)} g_m!} \prod_{k=1}^{j} \binom{\ell - \sum_{i=1}^{k-1} \ell_i}{\ell_k}^2 \frac{(\ell_k!)^2}{2\ell_k}. \tag{41}$$

Then, x_ℓ is obtained by summing (41) over all special partitions of the integer ℓ to yield

$$x_\ell = \sum_{((\ell_1,\dots,\ell_j))} \frac{1}{\prod_{m=1}^{\kappa(\ell_1,\dots,\ell_j)} g_m!} \prod_{k=1}^{j} \binom{\ell - \sum_{i=1}^{k-1} \ell_i}{\ell_k}^2 \frac{(\ell_k!)^2}{2\ell_k}. \tag{42}$$

The simplification of the factors $(\ell_k!)^2$ yields the expression stated by this lemma. ∎

Lemma 5: Consider bipartite graphs defined as follows: ℓ left vertices, ℓ right vertices, all left vertices have degree 2 except one vertex of degree 1, all right vertices have degree 2 except one vertex of degree 1, and finally no length-2 cycles are allowed. For $\ell \geq 3$, the total number y_ℓ of such bipartite graphs is

$$y_\ell = \ell^2 \cdot \left((2\ell - 1) \cdot x_{\ell-1} + (\ell - 1)^2 \cdot x_{\ell-2}\right), \tag{43}$$

where x_ℓ is determined via Lemma 4 and $x_1 = 0$.

Proof: Let the first $\ell{-}1$ left vertices and the first $\ell{-}1$ right vertices be of degree 2. There exists two ways to complete this bipartite graph such that the two remaining vertices have degree 1.

- Each of the $x_{\ell-1}$ sub-graphs has $2(\ell{-}1)$ edges. Break one edge into two edges and connect them to the remaining left and right vertices, the number of such graphs is $2(\ell - 1)x_{\ell-1}$. Another set of $x_{\ell-1}$ bipartite graphs is built by directly connecting the last two vertices together without breaking any edge in the upper sub-graph. Now, we find $2(\ell - 1)x_{\ell-1} + x_{\ell-1} = (2\ell - 1)x_{\ell-1}$ bipartite graphs.
- Fix a vertex among the $\ell{-}1$ upper left vertices and fix one among the $\ell - 1$ upper right vertices ($(\ell - 1)^2$ choices). Consider a length-2 cycle including these two vertices. One edge of this cycle can be broken into two edges and then attached to the degree-1 vertices at the bottom. The remaining $\ell - 2$ left and right vertices may involve $x_{\ell-2}$ sub-graphs. Consequently, the number of graphs in this second case is $(\ell - 1)^2 x_{\ell-2}$.

The total number of bipartite graphs enumerated in the above cases is

$$(2\ell - 1)x_{\ell-1} + (\ell - 1)^2 x_{\ell-2}. \tag{44}$$

Finally, the degree-1 left vertex has ℓ choices and so has the degree-1 right vertex. The number of graphs in (44) should be multiplied by ℓ^2. ∎

We make no claims about a possible generalization of Lemma 4 and Lemma 5 to finite bipartite graphs with higher vertex degrees. As mentioned before, for general degree distributions, results on enumeration of asymptotic bipartite graphs were published by Brendan McKay and his co-authors [15], [17]. Table I shows the number of special partitions for $\ell = 2 \dots 32$. The number of standard partitions (the partition function) can be found by a recursion resulting from the pentagonal number theorem [16]. To our knowledge, there exists no such recursion for special partitions. The number of bipartite graphs under the assumptions of Lemma 4 and Lemma 5 is found in Table II for a graph half-size up to 8. Finally, we are ready to state and prove the first theorem on stopping sets enumeration.

1, 1, 2, 2, 4, 4, 7, 8, 12, 14, 21, 24, 34, 41, 55, 66, 88, 105, 137, 165, 210, 253, 320, 383, 478, 574, 708, 847, 1039, 1238, 1507

Table I: Sequence of the number of special partitions of the integer ℓ, for $\ell = 2 \dots 32$. Special partitions are described in Definition 10. The sequence for standard partitions can be found in [59].

ℓ	2	3	4	5	6	7	8
x_ℓ	1	6	90	2040	67950	3110940	187530840
y_ℓ	0	45	816	22650	888840	46882710	3199593600

Table II: Number of bipartite graphs not including length-2 cycles from Lemma 4 and Lemma 5.

In the sequel, the open interval between two real numbers a and b will be denoted $]a, b[$,

$$]a, b[= \{x \in \mathbb{R} : a < x < b\}.$$

Theorem 2: Let C_P be a product code $[n_1, k_1, d_1]_q \otimes [n_2, k_2, d_2]_q$ built from row and column MDS component codes, where the alphabet size q is greater than $\max(n_1, n_2)$. Let τ_w be the number of stopping sets of size w. We write $\tau_w = \tau^a + \tau^b$, where τ^a counts obvious stopping sets and τ^b counts non-obvious stopping sets. Under (type-II) iterative algebraic decoding and for $d_1 = d_2 = d \geq 2$, stopping sets are characterized as follows:

- For $w < d^2$,
$$\tau^a = \tau^b = 0.$$
- For $w = d^2$,
$$\tau^a = \binom{n_1}{d}\binom{n_2}{d}, \quad \tau^b = 0.$$
- For $w \in]d^2, d(d + 1)[$,
$$\tau^a = \tau^b = 0.$$

- For $w = d(d+1)$,

$$\tau^a = \binom{n_1}{d}\binom{n_2}{d+1} + \binom{n_1}{d+1}\binom{n_2}{d},$$
$$\tau^b = (d+1)!\binom{n_1}{d+1}\binom{n_2}{d+1}.$$

- For $w \in \,]d(d+1), d(d+2)[$.
Let us write $w = d^2 + d + \lambda$, where $\lambda \in [1, d-1]$.

$$\tau^a = 0,$$
$$\tau^b = (d+1-\lambda)!\binom{d+1}{\lambda}^2\binom{n_1}{d+1}\binom{n_2}{d+1}.$$

- For $w = d(d+2)$,

$$\tau^a = \binom{n_1}{d}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d},$$
$$\tau^b = (d+1)^2\binom{n_1}{d+1}\binom{n_2}{d+1}$$
$$+ \sum_{2r_0+r_1=d}\binom{d+1}{r_0}\binom{d+1-r_0}{r_1}\frac{(d+2)!}{2^{r_2}}$$
$$\left[\binom{n_1}{d+1}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d+1}\right]$$
$$+ x_{d+2}\binom{n_1}{d+2}\binom{n_2}{d+2},$$

where $\sum_{2r_0+r_1=d}$ is a summation over r_0 and r_1, both being non-negative and satisfying $2r_0 + r_1 = d$, $r_2 = d + 1 - r_0 - r_1$, and x_{d+2} is determined from Lemma 4.

- For $w = (d+1)(d+1)$

$$\tau^a = \binom{n_1}{d+1}\binom{n_2}{d+1},$$
$$\tau^b = \sum_{2r_0+r_1=d+1}\binom{d+1}{r_0}\binom{d+1-r_0}{r_1}\frac{(d+2)!}{2^{r_0}}$$
$$\left[\binom{n_1}{d+1}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d+1}\right]$$
$$+ y_{d+2}\binom{n_1}{d+2}\binom{n_2}{d+2},$$

where y_{d+2} is determined from Lemma 5.

Proof: For w satisfying $d^2 \leq w \leq (d+1)^2$, the admissible size of $\mathcal{R}(\mathcal{S})$ varies from d^2 up to $(d+2)^2$ as given by Lemma 2. All cases stated in the theorem shall use the following sequence of $\mathcal{R}(\mathcal{S})$ listed in the order of increasing size $\ell_1\ell_2$: d^2, $d(d+1)$, $d(d+2)$, $(d+1)^2$, $(d+1)(d+2)$, and $(d+2)^2$. For these rectangular supports, the stopping set weight also has six cases to be considered, where w takes the following values (or ranges) in increasing order: $w = d^2$, $w \in]d^2, d(d+1)[$, $w = d(d+1)$, $w \in]d(d+1), d(d+2)[$, $w = d(d+2)$, and $w = (d+1)^2$.

- The case $w < d^2$.
Consider a stopping set of size $w < d^2$. Its rectangular support $\mathcal{R}(\mathcal{S})$ has size $\ell_1\ell_2 \geq w$. All columns should have a weight greater than or equal to d, we find that $w \geq d\ell_2$. Similarly, all rows must have a weight greater than or equal to d, then $w \geq d\ell_1$. By combining the two inequalities, we find $w^2 \geq d^2\ell_1\ell_2 \geq d^2 w$, so we get $w \geq d^2$ which is a contradiction unless these stopping sets do not exist, i.e. $\tau_w = 0$ for $w < d^2$ under type II iterative decoding.

- The case $w = d^2$.
We use similar inequalities as in the previous case. We have $w = d^2 \geq d\ell_2$ because column decoding must fail. We obtain $\ell_2 \leq d$. In a symmetric way, $w = d^2 \geq d\ell_1$ because row decoding must fail. We obtain $\ell_1 \leq d$. But $\mathcal{R}(\mathcal{S})$ cannot be smaller than $\mathcal{S}$, i.e. we get $\ell_1 = d$ and $\ell_2 = d$. We just proved that all stopping set of size d^2 are obvious. Their number is given by choosing d rows out of n_1 and d columns out of n_2.

- The case $d^2 < w < d(d+1)$.
Given that $\ell_1\ell_2 \geq w > d^2$, we get $\ell_1 \geq d$ and $\ell_2 \geq d$ since the support $\mathcal{R}(\mathcal{S})$ is larger than a $d \times d$ rectangle, the latter being the smallest stopping set as proven in the previous case. Take $\ell_1 = d$, then $\ell_2 \geq d+1$ because $w > d^2$. The weight of each column must be at least d giving us $w \geq d\ell_2 \geq d(d+1)$, which is a contradiction unless $\tau_w = 0$. For $\ell_1 > d$, the same arguments hold.

- The case $w = d(d+1)$.
  - The smallest $\mathcal{R}(\mathcal{S})$ is $d \times (d+1)$ or $(d+1) \times d$. According to Lemma 3, we have $\beta = 0$. All these stopping sets are obvious. Their number is
  $$\binom{n_1}{d}\binom{n_2}{d+1} + \binom{n_1}{d+1}\binom{n_2}{d}.$$
  - $\mathcal{R}(\mathcal{S})$ has size $d(d+2)$. Each column must have at least d erasures. Then $\mathcal{S}$ can only be obvious with weight $d(d+2)$ which contradicts $w = d(d+1)$. Hence, this size of rectangular support yields no stopping sets, $\tau_w = 0$ in this sub-case.
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+1)$. Let β be the number of zeros in $\mathcal{R}(\mathcal{S})$, $\beta = (d+1)^2 - w = d+1$. All these stopping sets are found by considering the $(d+1)!$ permutations where a unique 0 is placed per row and per column. Then, the binomial coefficient must be multiplied by $(d+1)!$ which yields the τ^b announced in the theorem for $w = d(d+1)$.
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+2)$. The number of zeros is $\beta = (d+1)(d+2) - w = 2d+2$. Then $\beta > d+2 = (\ell_1 - d)\ell_2$ which contradicts Lemma 3. We get $\tau_w = 0$ in this sub-case. The same arguments are valid for larger rectangles.

- The case $d(d+1) < w < d(d+2)$.
Let us write $w = d^2 + d + \lambda$, where $\lambda \in [1, d-1]$. We consider below three sub-cases corresponding to admissible sizes of $\mathcal{R}(\mathcal{S})$.
  - The smallest $\mathcal{R}(\mathcal{S})$ is $d \times (d+2)$ or $(d+2) \times d$. Take the rectangle of size $d \times (d+2)$. Each column must have at least d erasures. Then $\mathcal{S}$ can only be obvious with weight $d(d+2)$ which is outside the range for w in this case. Hence, this size of the rectangular

  support yields no stopping sets, $\tau_w = 0$ in this sub-case.
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)\times(d+1)$. The number of zeros is $\beta = (d+1)^2 - w = d+1-\lambda$, where $\beta \in [2, d]$. Put the zeros in $\mathcal{R}(\mathcal{S})$ not exceeding one per column and not exceeding one per row. The enumeration of these stopping sets is given by selecting the β rows and the β columns, then filling all $\beta \times \beta$ permutation matrices in the zero positions. Hence, given that $\binom{d+1}{\beta} = \binom{d+1}{\lambda}$, we get for this sub-case

    $$\tau_w = \beta! \binom{d+1}{\lambda}^2 \binom{n_1}{d+1}\binom{n_2}{d+1}.$$

    All corresponding stopping sets are not obvious (the rank is greater than 1).
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+2)$. The number of zeros is $\beta = (d+1)(d+2) - w = 2d+2-\lambda \in [d+3, 2d+1]$. Then $\beta > d+2 = (\ell_1 - d)\ell_2$ which contradicts Lemma 3. In a similar way, it can be proven that $\tau_w = 0$ in the sub-case $\mathcal{R}(\mathcal{S})$ with size $(d+2)^2$.

- The case $w = d(d+2)$.
  The admissible rectangular support can have four sizes: $d(d+2)$, $(d+1)(d+1)$, $(d+1)(d+2)$, and $(d+2)(d+2)$.
  - $\mathcal{R}(\mathcal{S})$ has size $d(d+2)$. According to Lemma 3, we have $\beta = 0$. All these stopping sets are obvious. Their number is

    $$\binom{n_1}{d}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d}.$$
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+1)$. We have $\beta = 1$. The number of these stopping sets is

    $$(d+1)^2 \binom{n_1}{d+1}\binom{n_2}{d+1}.$$
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+2)$. The number of zeros is $\beta = d+2$. Each column must have a unique zero and each row cannot have more than two zeros. Let r_i be the number of rows containing i zeros, $i = 0, 1, 2$. Then $r_0 + r_1 + r_2 = d+1$ and $\beta = 2r_2 + r_1$, so the constraint is $2r_0 + r_1 = d$. Given a stopping set satisfying this constraint, a permutation can be applied on the $(d+2)$ columns to create another stopping set. But a row with two zeros creates two identical columns, so the number of stopping sets should be divided by 2^{r_2}, where $r_2 = d+1-r_0-r_1$. The number of stopping sets in this sub-case is

    $$\sum_{2r_0+r_1=d} \binom{d+1}{r_0}\binom{d+1-r_0}{r_1}\frac{(d+2)!}{2^{r_2}} \left[\binom{n_1}{d+1}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d+1}\right].$$
  - $\mathcal{R}(\mathcal{S})$ has size $(d+2)(d+2)$. We have $\beta = 2d+4$ reaching the upper bound in Lemma 3. $\mathcal{R}(\mathcal{S})$ must have two zeros in each column and two zeros in each row. A first group of these stopping sets can be enumerated by building $\mathcal{R}(\mathcal{S})$ with two zero length-$(d+2)$ diagonals (to be folded if not the main diagonal) and then applying all row and column permutations. This generates all Hamiltonian bipartite graphs with $d+2$ left vertices and $d+2$ right vertices, their number is

    $$\frac{((d+2)!)^2}{2(d+2)},$$

    as known from Lemma 4. In fact, the full exact enumeration of stopping sets in this case is already made by Lemma 4 and its proof, just take $\ell = d+2$. Then, in this sub-case, the number of stopping sets is given by

    $$x_{d+2}\binom{n_1}{d+2}\binom{n_2}{d+2}.$$

- The case $w = (d+1)(d+1)$.
  The admissible rectangular support can have three possible sizes $(d+1)(d+1)$, $(d+1)(d+2)$, and $(d+2)(d+2)$.
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+1)$. We have $\beta = 0$, i.e. $\mathcal{R}(\mathcal{S}) = \mathcal{S}$. The number of these obvious stopping sets is

    $$\binom{n_1}{d+1}\binom{n_2}{d+1}.$$
  - $\mathcal{R}(\mathcal{S})$ has size $(d+1)(d+2)$. We have $\beta = d+1$. A column of $\mathcal{R}(\mathcal{S})$ should contain at most one zero and a row should contain at most two zeros. Let r_i be the number of rows containing i zeros, $i = 0, 1, 2$. Then $r_0+r_1+r_2 = d+1$ and $\beta = 2r_2+r_1$, so the constraint is $2r_0 + r_1 = d+1$. Given a stopping set satisfying this constraint, a permutation can be applied on the $(d+2)$ columns to create another stopping set. The number of stopping sets in this sub-case is

    $$\sum_{2r_0+r_1=d+1} \binom{d+1}{r_0}\binom{d+1-r_0}{r_1}\frac{(d+2)!}{2^{r_2}} \left[\binom{n_1}{d+1}\binom{n_2}{d+2} + \binom{n_1}{d+2}\binom{n_2}{d+1}\right],$$

    where $r_2 = r_0$.
  - $\mathcal{R}(\mathcal{S})$ has size $(d+2)(d+2)$. $\beta = 2d+3$ which is less than the upper bound in Lemma 3. These stopping sets are equivalent to bipartite graphs considered in Lemma 5. Then, in this sub-case, the number of stopping sets is given by

    $$y_{d+2}\binom{n_1}{d+2}\binom{n_2}{d+2}.$$

■

From the proof of Theorem 2, in the case $w = d(d+2)$ with a $(d+1)\times(d+2)$ rectangular support, the enumeration of stopping sets is directly converted into enumeration of trivial bipartite graphs defined by: a- ℓ left vertices and a left degree 0, 1, or 2, and b- $\ell+1$ right vertices all of degree 1. Similarly, the proof for the case $w = d(d+2)$ with a $(d+1)\times(d+2)$ rectangular support is directly related to the enumeration of bipartite graphs with one edge less.

For $d_1 \neq d_2$, a very long theorem on stopping sets enumeration is found in the arXiv version of this paper. In the current paper, we refer to both theorems in the cases $d_1 = d_2$ and $d_1 \neq d_2$ when citing Theorem 2, From a stopping set perspective, our enumeration theorems match Tolhuizen's results on weight distribution for a weight less than $d_1 d_2 + d_2$ [64]. Our theorems found stopping sets that are only obvious for w in the range $[d_1 d_2, d_1 d_2 + d_2[$. For any weight w, there exists an equivalence in support between codewords and obvious stopping sets (thanks to Proposition 3, to be presented in Section VI-B). Trivial lower and upper bounds of the number of obvious weight-w product code codewords are

$$(q-1)\tau_w \leq A_w \leq \mathbb{1}_{\{\tau_w \neq 0\}} A_w.$$

For non-obvious stopping sets and non-obvious codewords, establishing a clear relationship is still an open problem. This is directly related to solving the weight enumeration beyond $d_1 d_2 + \max(d_1, d_2)$. In the special case $d_1 = d_2 = d$, Sendrier gave upper bounds of the number of erasure patterns for a weight up to $d^2 + 2d - 1$ [58].

VI. Code performance in presence of erasures

Iterative decoding performance of $C_P = C_1 \otimes C_2$ is studied in presence of channel erasures, with and without edge coloring. The iterative decoder makes row and column iterations where the component decoder of C_i can be an algebraic erasure-filling decoder (limited by $d_i - 1$) or a maximum-likelihood decoder of C_i. As stated in Section V-A, type II and type III stopping sets are identical because the non-binary codes C_1 and C_2 are MDS. The word error probability of the iterative decoder is denoted by $P_{ew}^{\mathcal{G}}$. The product code can also be decoded via an ML decoder, i.e. maximum likelihood decoding of C_P based on a Gaussian reduction of its parity-check matrix. The word error probability under ML decoding of C_P is denoted by P_{ew}^{ML}.

A. Block erasures

Consider the block-erasure channel $CEC(q, \epsilon)$. The N symbols of a codeword are partitioned into M blocks, each block contains symbols associated to edges in $\mathcal{G}$ with the same color. The $CEC(q, \epsilon)$ channel erases a block with a probability ϵ. The block is correctly received with a probability $1-\epsilon$. Erasure events are independent from one block to another. We say that a *color is erased* if the associated block of N/M symbols is erased. Assume that $\mathcal{G}$ is endowed with a double-diversity edge coloring ϕ (i.e. $L(\phi) = 2$) as defined in Corollary 1. Then, on the block-erasure channel $CEC(q, \epsilon)$, for a rate satisfying

$$1 - \frac{2}{M} < R \leq 1 - \frac{1}{M}, \tag{45}$$

we have

$$\epsilon^2 \;\leq\; P_{ew}^{ML} \;\leq\; P_{ew}^{\mathcal{G}} \;\leq\; \sum_{i=2}^{M} \binom{M}{i} \epsilon^i (1-\epsilon)^{M-i}. \tag{46}$$

Since ϕ has a double diversity, there exist two colors among the M colors such that the iterative decoder must fail if both colors are erased. This explains the upper bound of $P_{ew}^{\mathcal{G}}$ in (46). The upper bound is valid for any rate less than the maximal achievable rate for double diversity, i.e. $1 - \frac{1}{M}$. Now, since $R > 1 - \frac{2}{M}$, the ML decoder for C_P cannot attain a diversity $L = 3$ otherwise the block-fading/block-erasure Singleton bound would be violated. Consequently, the ML decoder of C_P can only reach $L = 2$ and so there exists a pair of erased colors that cannot be solved by the ML decoder. This explains the lower bound in (46). The reader can easily verify that

$$\lim_{\epsilon \to 0} \frac{\log P_{ew}^{ML}}{\log \epsilon} = \lim_{\epsilon \to 0} \frac{\log P_{ew}^{\mathcal{G}}}{\log \epsilon} = L = 2. \tag{47}$$

The slope of P_{ew} versus the erasure probability ϵ in a double-logarithmic scale is equal to 2. Under the stated constraint on R, the upper bound in (46) is the exact expression of the outage probability on a block-erasure channel valid for q-ary codes with asymptotic length [27]. For double-diversity edge colorings found by DECA in Examples 3 and 4, $P_{ew}^{\mathcal{G}}$ equals its upper bound in (46). These examples achieve the outage probability although a code may perform better than the outage probability at finite length. For these colorings where $M = 4$, the error probability on $CEC(q, \epsilon)$ behaves like $P_{ew}^{\mathcal{G}} = 6\epsilon^2 + O(\epsilon^3)$. One possible interpretation of this behavior is: the optimization of $\eta(\phi)$ (equivalent in some sense to minimizing $\rho(\phi)$) pushed the performance of edge colorings found by the DECA algorithm as far as possible from the lower bound ϵ^2. As can be observed in Figures 6 and 7, all rows and all columns include the four colors. When any two colors out of four are erased, the iterative decoder will completely fail without correcting a single supersymbol. A double-diversity edge coloring guarantees that all stopping sets are covered by at least two colors but it cannot cover all stopping sets with three colors or more otherwise we get $L = 3$ which contradicts $R > 1 - \frac{2}{M}$. Fortunately, these product codes are diversity-wise MDS and the second code in Example 4 has the maximal coding rate for double diversity. In the sequel, we will see that these codes also perform well in presence of independent erasures.

B. Independent erasures

Consider the i.i.d. erasure channel $SEC(q, \epsilon)$. The N symbols of a codeword are independently erased by the channel. A symbol is erased with a probability ϵ and is correctly received with a probability $1 - \epsilon$. Edge coloring has no effect on the performance of C_P on the $SEC(q, \epsilon)$ channel. Before studying the performance on the $SEC(q, \epsilon)$, following Examples 6 & 7 and Theorem 2, we state an obvious result about obvious stopping sets in the following proposition.

Proposition 3: Let $C_P = C_1 \otimes C_2$ be a product code with non-binary MDS components. All obvious stopping sets are supports of product code codewords.

Proof: Consider an $\ell_1 \times \ell_2$ obvious stopping set. Its rectangular support is $\mathcal{R}(\mathcal{S}) = \mathcal{R}_1(\mathcal{S}) \times \mathcal{R}_2(\mathcal{S})$. We have $\ell_1 \geq d_1$ and $\ell_2 \geq d_2$. From Proposition 2, there exists a column codeword $x = (x_1, x_2, \ldots, x_{n_1}) \in C_1$ of weight ℓ_1 with support $\mathcal{R}_1(\mathcal{S}) \times \{j_1\}$, where $j_1 \in \mathcal{R}_2(\mathcal{S})$. Similarly, there exists a row codeword $y = (y_1, y_2, \ldots, y_{n_2}) \in C_2$ of weight ℓ_2 with support $\{i_1\} \times \mathcal{R}_2(\mathcal{S})$, where $i_1 \in \mathcal{R}_1(\mathcal{S})$.

Now, the Kronecker product of x and y satisfies $\mathcal{X}(x \otimes y) = \mathcal{S}$. ■

Corollary 5: Consider a product code $C_P = C_1 \otimes C_2$ with non-binary MDS component codes. Assume the symbols of C_P are transmitted over a $SEC(q, \epsilon)$ channel. Then, for $\epsilon \ll 1$, the error probabilities satisfy $P_{ew}^{\mathcal{G}} \sim P_{ew}^{ML}$.

Proof: On the $SEC(q, \epsilon)$, the word error probabilities are given by [56],

$$P_{ew}^{ML} = \sum_{i=d_1 d_2}^{N} \Psi_i(ML) \epsilon^i (1-\epsilon)^{N-i}, \tag{48}$$

where $\Psi_i(ML)$ is the number of weight-i erasure patterns covering a product code codeword, and

$$P_{ew}^{\mathcal{G}} = \sum_{i=d_1 d_2}^{N} \Psi_i(\mathcal{G}) \epsilon^i (1-\epsilon)^{N-i}, \tag{49}$$

where $\Psi_i(\mathcal{G})$ is the number of weight-i erasure patterns covering a stopping set. Of course, here we refer to stopping sets in the non-compact graph $\mathcal{G}$, i.e. in the $n_1 \times n_2$ product code matrix. Next, since N is fixed (asymptotic length analysis is not considered in this paper) we write $P_{ew}^{ML} = \Psi_{d_1 d_2}(ML) \epsilon^{d_1 d_2} + o(\epsilon^{d_1 d_2})$ and $P_{ew}^{\mathcal{G}} = \Psi_{d_1 d_2}(\mathcal{G}) \epsilon^{d_1 d_2} + o(\epsilon^{d_1 d_2})$. From Proposition 3, we get the equality $\Psi_{d_1 d_2}(\mathcal{G}) = \Psi_{d_1 d_2}(ML)$ and so we obtain $\lim_{\epsilon \to 0} P_{ew}^{\mathcal{G}} / P_{ew}^{ML} = 1$. ■

The erasure patterns can be decomposed according to the size of the covered stopping set. The coefficient $\Psi_i(\mathcal{G})$ becomes $\Psi_i(\mathcal{G}) = \sum_{w=d_1 d_2}^{i} \Psi_{i,w}(\mathcal{G})$, where $\Psi_{i,w}(\mathcal{G})$ is the number of weight-i patterns covering a stopping set of size w. It is clear that $\Psi_{w,w}(\mathcal{G}) = \tau_w$. For small $i - w$, $\Psi_{i,w}(\mathcal{G})$ can be approximated by $\sum_{\mathcal{A}} \binom{N-\mathcal{A}}{i-w} \tau_{w,\mathcal{A}}$, where $\tau_{w,\mathcal{A}}$ is the number of stopping sets of size w having $|\mathcal{R}(\mathcal{S})| = \mathcal{A}$. For $w \leq d_1 d_2 + d_1 + d_2 + 1$, the area $\mathcal{A}$ is bounded from above by the product $\ell_1^0 \times \ell_2^0$ from Lemma 2. Numerical evaluations of $\Psi_i(\mathcal{G})$ are tractable for very short codes ($N \leq 25$) and become very difficult for codes of moderate size and beyond, e.g. $N = 144$ and $N = 224$ for the $[12, 10]^{\otimes 2}$ and the $[14, 12] \otimes [16, 14]$ codes respectively. For this reason, expressions (48) and (49) are not practical to predict the $SEC(q, \epsilon)$ performance of product codes with significant characteristics.

For $P_{ew}^{\mathcal{G}}$, thanks to Theorem 2, a union bound can be easily established. Indeed, we have

$$\begin{aligned} P_{ew}^{\mathcal{G}} &= Prob(\exists \mathcal{S}\ covered) \\ &\leq \sum_{w} Prob(\exists \mathcal{S} : |\mathcal{S}| = w, \mathcal{S}\ covered), \end{aligned}$$

leading to

$$P_{ew}^{\mathcal{G}} \leq P^U(\epsilon) = \sum_{w=d_1 d_2}^{N} \tau_w \epsilon^w. \tag{50}$$

From Theorem 2, the union bound $P^U(\epsilon)$ for the $[12, 10, 3]_q^{\otimes 2}$ product code is

$$\begin{aligned} P^U(\epsilon) = &48400\epsilon^9 + 6098400\epsilon^{12} + 23522400\epsilon^{13} + 17641800\epsilon^{14} \\ &+ 1754335440\epsilon^{15} + 9126691200\epsilon^{16} + o(\epsilon^{16}). \end{aligned}$$

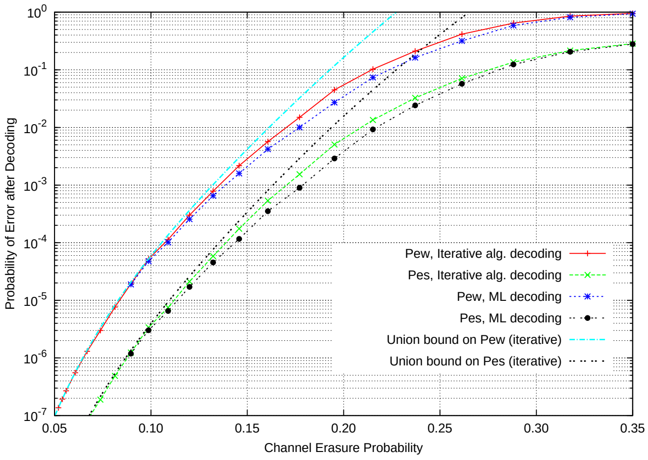


Figure 11: Product code $[12, 10]_q^{\otimes 2}$, no edge coloring. Word and symbol error rate performance for iterative decoding versus its union bound and ML decoding.

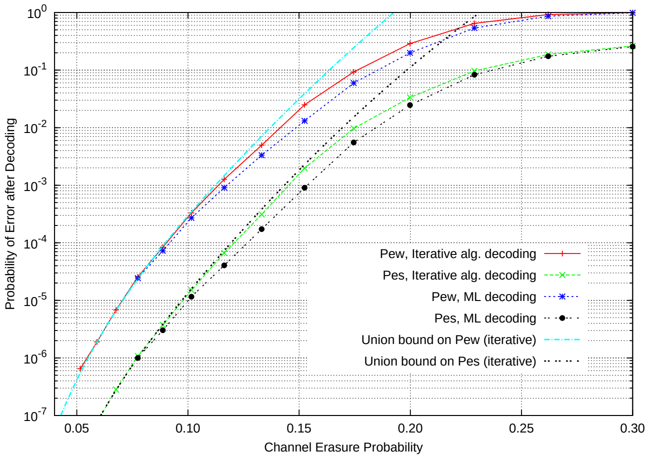


Figure 12: Product code $[14, 12]_q \otimes [16, 14]_q$, no edge coloring. Word and symbol error rate performance for iterative decoding versus its union bound and ML decoding.

The performance of this code on the $SEC(q, \epsilon)$ channel is shown in Figure 11. We used the standard finite field of size $q = 256$. The union bound for the symbol error probability $P_{es}^{\mathcal{G}}$ is derived by weighting the summation term in (50) with w/N, i.e. $P_{es}^{\mathcal{G}} \leq \sum_{w=d_1 d_2}^{N} \frac{w}{N} \tau_w \epsilon^w$. As observed in the plot of Figure 11, the union bound is sufficiently tight. Furthermore, the performance of the iterative algebraic row-column decoder is very close to that of ML decoding in the whole range of ϵ. For small ϵ, the curves are superimposed as predicted by Corollary 5.

The union bound $P^U(\epsilon)$ for the $[14, 12, 3]_q \otimes [16, 14, 3]_q$ product code is

$$\begin{aligned} P^U(\epsilon) = &203840\epsilon^9 + 44946720\epsilon^{12} + 174894720\epsilon^{13} \\ &+ 131171040\epsilon^{14} + 17839261440\epsilon^{15} \\ &+ 126887941180\epsilon^{16} + o(\epsilon^{16}). \end{aligned}$$

The performance of this code on the $SEC(q, \epsilon)$ channel is shown in Figure 12. Similar to the previous code, the union bound is tight enough and iterative decoding performs very close to ML decoding. Finally, let us interpret these results from a finite-length information theoretical point of view [45]. The $SEC(q, \epsilon)$ of Shannon capacity $\log_2(q)(1-\epsilon)$ behaves exactly like a $BEC(\epsilon)$ of capacity $(1-\epsilon)$ but erasures in the SEC occur at the symbol level instead of the binary digit level. Finite-regime BEC bounds from [45] are directly applicable to our product codes over the $SEC(q, \epsilon)$. The BEC channel dispersion is $V = \epsilon(1-\epsilon)$ and its maximal achievable rate is given by [45], Theorem 53,

$$R = (1-\epsilon) - \sqrt{\frac{V}{n}} Q^{-1}(P_{ew}) + O(\frac{1}{n}), \tag{51}$$

where n is the code length, $Q(x)$ is the Gaussian tail function, ϵ is the channel erasure probability, and P_{ew} is the target word error probability. The next table shows how good is the proposed product code based on MDS components.

	Coding Rate R for $\epsilon = 0.15$	Erasure Prob. ϵ for $R = 0.75$
Polyanskiy-Poor-Verdú	$0.794 : P_{ew} = 1.0 \cdot 10^{-2}$	0.189
$[14, 12]_q \otimes [16, 14]_q$	$0.750 : P_{ew} = 1.0 \cdot 10^{-2}$	0.150
Regular-$(3, 12)$ LDPC	$0.750 : P_{ew} = 2.9 \cdot 10^{-2}$	0.135

Table III: Finite-length performance of the $[14, 12]_q \otimes [16, 14]_q$ product code. The value of ϵ in the third column is given for $P_{ew} = 10^{-2}$ at all rows.

C. Unequal probability erasures

In communication and storage systems, erasure events of unequal probabilities may occur. In order to observe the effect of a double-diversity coloring on the performance in multiple erasure channels, we define the $SEC(q, \{\epsilon_i\}_{i=1}^{M})$. On this channel, symbol erasure events are independent but the probability of erasing a symbol is ϵ_i if it is associated to an edge in $\mathcal{G}$ with color $\phi(e) = i$. The union bound is easily modified to get

$$P_{ew}^{\mathcal{G}} \leq P^U(\epsilon_1, \ldots, \epsilon_M), \tag{52}$$

where

$$P^U(\epsilon_1, \ldots, \epsilon_M) = \sum_{w=d_1 d_2}^{N} \sum_{\substack{w_1, \ldots, w_M \\ : \sum_i w_i = w}} \tau(w_1, \ldots, w_M) \prod_{i=1}^{M} \epsilon_i^{w_i}. \tag{53}$$

The coefficient $\tau(w_1, \ldots, w_M)$ is the number of stopping sets of size $w = \sum_{i=1}^{M} w_i$, where i symbol edges have color i, $i = 1 \ldots M$. Clearly, the coefficients $\tau(w_1, \ldots, w_M)$ depend on the edge coloring ϕ. For double-diversity colorings and $M \geq 2$, these coefficients satisfy the following property:
For any stopping set $\mathcal{S}$ such that $|\mathcal{S}| = w$, $\tau(w_1, \ldots, w_M)$ does exist for $\sum_{i=1}^{M} w_i = w$ and $w_i > 0$ only, i.e. no weak compositions of w are authorized by ϕ.
Hence, the product code should perform well if one of the ϵ_i is close to 1 and the remaining ϵ_i are small enough. The extreme case is true thanks to double diversity yielding $P^U(0^{M-1}, 1^1) = 0$, where $(0^{M-1}, 1^1)$ represents all vectors with all positions at 0 except for one position set to 1. Figure 13 shows the performance of $[12, 10]_q^{\otimes 2}$ on the $SEC(q, \{\epsilon_i\}_{i=1}^{M})$ channel with $M = 4$ colors. The edge coloring is the double-diversity coloring produced by the DECA algorithm and drawn in Figure 6. The expression of $P^U(\epsilon_1, \ldots, \epsilon_M)$ is determined by stopping sets enumeration as in Theorem 2. Details are omitted and the very long expression of $P^U(\epsilon_1, \ldots, \epsilon_M)$ is not shown. The special case $\epsilon_1 = \epsilon_2 = \epsilon_3$ is considered and the performance is plotted as a function of ϵ_4. For a fixed ϵ_1, double diversity dramatically improves the performance with respect to ϵ_4.

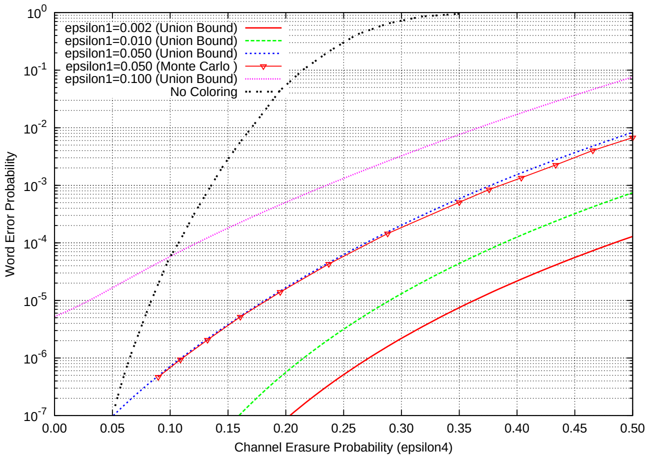


Figure 13: Product code $[12, 10]_q^{\otimes 2}$ with double-diversity edge coloring. Word error rate performance versus ϵ_4, for iterative decoding on the $SEC(q, \{\epsilon_i\}_{i=1}^{4})$ channel with $\epsilon_1 = \epsilon_2 = \epsilon_3$.

VII. Conclusions

Non-binary product codes with MDS components are studied in this paper in the context of iterative row-column algebraic decoding. Channels with both independent and block erasures are considered. The rootcheck concept and associated double-diversity edge colorings were described after introducing a compact graph representation for product codes. For solving erased symbols, an upper bound of the number of decoding iterations is given as a function of the graph size and the color palette size M. We proposed a differential evolution edge coloring algorithm to design colorings with a large population of minimal rootcheck order symbols. The complexity of this algorithm per iteration is $o(M^{\aleph})$, where $\aleph$ is the differential evolution parameter. Then, stopping sets are defined in the context of MDS components and a relationship is established with the graph representation of the product code. A full characterization of these stopping sets is given up to a weight $(d_1+1)(d_2+1)$. The performance of MDS-based product codes with and without double-diversity coloring is analyzed. In addition, ML and iterative row-column algebraic decoding are proven to coincide at small channel erasure probability. Original results found in this paper are listed in Section I-B.

A complete enumeration of product code codewords is still an open problem in coding theory. Following the enumeration of bipartite graphs in Section V-D (see also Table I) and following the DECA algorithm that aims at improving $\eta(\phi)$ in Section IV-B, two open problems can be stated.

• In number theory. There exists no recursive or closed form expression for the special partition function, i.e. the number of special partitions of an integer. Also, in a way similar to the Hardy-Ramanujan formula, the asymptotic behavior is unknown for the number of special partitions. Special partitions are introduced in Definition 10.

• In graph theory and combinatorics. Consider a matrix of size $H \times W$ and a coloring palette of size M. For simplicity, assume that $H \cdot W$ is multiple of M. A matrix entry is called *edge*. A color is assigned to each edge in the matrix. All M colors are equally used. A matrix edge/entry (i, j) of color c is said to be *good* if it is the unique entry with color c either on row i or on column j. The number of good entries is denoted by $\eta(\phi)$, see also (19). Given the matrix height H, width W, and the palette size M, find the maximum achievable number of good entries $\eta(\phi)$ over the set of all edge colorings ϕ. A simpler problem would be to find an upper bound of $\eta(\phi)$.

Acknowledgment

The work of Joseph J. Boutros was supported by the Qatar National Research Fund (QNRF), a member of Qatar Foundation, under NPRP project 5-401-2-161 on layered coding. The authors would like to thank Dr. Mireille Sarkiss, from CEA-LIST Paris, for her precious support. The authors are also grateful to the anonymous reviewers for their valuable comments that helped in improving the paper presentation.

References

[1] N. Abramson, "Cascade decoding of cyclic product codes," *IEEE Trans. on Comm. Technology*, vol. 16, no. 3, pp. 398-402, June 1968.

[2] M. Alipour, O. Etesami, G. Maatouk, and A. Shokrollahi, "Irregular product codes," *IEEE Information Theory Workshop*, pp 197-201, Lausanne, Sept. 2012.

[3] D. Augot, M. El-Khamy, R.J. McEliece, F. Parvaresh, M. Stepanov, and A. Vardy, "Algebraic list decoding of Reed-Solomon product codes," *Algebraic and Combinatorial Coding Workshop*, pp. 210-213, Sept. 2006.

[4] E. Arıkan, "Channel polarization: A method for constructing capacity achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051-3073, July 2009.

[5] S.B. Balaji, G.R. Kini, and P.V. Kumar, "A bound on rate of codes with locality with sequential recovery from multiple erasures," Dec. 2016, preprint, arXiv:1611.08561v3.

[6] S. Benedetto and G. Montorsi, "Unveiling turbo-codes: some results on parallel concatenated coding schemes," *IEEE Trans. on Inf. Theory*, vol. 42, no. 2, pp. 409-428, March 1996.

[7] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. on Communications*, vol. 44, pp. 1261-1271, Oct. 1996.

[8] R.E. Blahut, *Algebraic codes for data transmission*, Cambridge University Press, 2003.

[9] B. Bollobás, *Modern graph theory*, Springer, 1998.

[10] J.J. Boutros, O. Pothier, and G. Zémor, "Generalized low density (Tanner) codes," *IEEE Intern. Conf. on Comm. (ICC)*, vol. 1, pp. 441-445, Vancouver, June 1999.

[11] J.J. Boutros, G. Zémor, A. Guillén i Fàbregas, and E. Biglieri, "Full-diversity product codes for block erasure and block fading channels," *Information Theory Workshop*, pp. 313-317, Porto, May 2008.

[12] J.J. Boutros, G. Zémor, A. Guillén I Fàbregas, and E. Biglieri, "Generalized low-density codes with BCH constituents for full-diversity near-outage performance," *IEEE Intern. Symp. on Inform. Theory (ISIT)*, pp. 787-791, Toronto, July 2008.

[13] J.J. Boutros, "Diversity and coding gain evolution in graph codes," *Information Theory and Applications Workshop*, pp. 34-43, San Diego, Feb. 2009.

[14] J.J. Boutros, A. Guillén i Fàbregas, E. Biglieri, and G. Zémor, "Low-density parity-check codes for nonergodic block-fading channels," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4286-4300, Sept. 2010.

[15] E.R. Canfield and B.D. McKay, "Asymptotic enumeration of integer matrices with constant row and column sums", Combinatorics, arXiv:math/0703600, revised June 2009.

[16] J.H. Conway and R.K. Guy. *The Book of Numbers*. New York: Springer-Verlag, pp. 94-96, 1996.

[17] C. Greenhill and B.D. McKay, "Asymptotic enumeration of sparse nonnegative integer matrices with specified row and column sums," Advances in Applied Mathematics, vol. 41, pp. 459-481, 2008, revised April 2012.

[18] C. Di, D. Proietti, I.E. Telatar, T.J. Richardson, and R.L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1570-1579, Jun. 2002.

[19] A.G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *IEEE Proceedings*, vol. 99, pp. 476-489, March 2011.

[20] P. Elias, "Error-free coding," *IRE Trans. Inf. Theory*, vol. 4, no. 4, pp. 29-39, Sept. 1954.

[21] M. El-Khamy and R.J. McEliece, "Iterative algebraic soft-decision list decoding of Reed-Solomon codes," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 481-490, March 2006.

[22] K.S. Esmaili, L. Pamies-Juarez, and A. Datta, "CORE: Cross-object redundancy for efficient data repair in storage systems," *IEEE Big Data*, pp. 246-254, Oct. 2013.

[23] D.F. Freeman and A.M. Michelson, "A two-dimensional product code with robust soft-decision decoding," *IEEE Trans. Comm.*, vol. 44, no. 10, pp. 1222-1226, Oct. 1996.

[24] R.G. Gallager, *Low-density parity-check codes*, Ph.D. thesis, Massachussets Institute of Technology Press, 1963.

[25] P. Gopalan, V. Guruswami, and P. Raghavendra, "List decoding tensor products and interleaved codes," *SIAM J. Comput.*, vol. 40, no. 5, pp. 1432-1462, Oct. 2011.

[26] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6925-6934, Nov. 2012.

[27] A. Guillén i Fàbregas, "Coding in the block-erasure channel," *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 5116-5121, Nov. 2006.

[28] A. Guillén i Fàbregas and G. Caire, "Coded modulation in the block-fading channel: Coding theorems and code construction," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, pp. 91-114, Jan. 2006.

[29] V. Guruswami and M. Sudan, "Improved decoding of Reed-Solomon and algebraic-geometry codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 17571767, June 1999.

[30] T. Helleseth, T. Klove, and O. Ytrehus, "Generalized Hamming weights of linear codes," *IEEE Trans. Inf. Theory*, vol. 38, no. 3, pp. 1133-1140, May 1992.

[31] J. Jiang, K.R. Narayanan, "Iterative soft decoding of Reed-Solomon codes," *IEEE Communications Letters*, vol. 8, no. 4, pp. 244-246, April 2004.

[32] R. Knopp and P.A. Humblet, "On coding for block fading channels," *IEEE Trans. Inf. Theory*, vol. 46, no. 1, pp. 189-205, Jan. 2000.

[33] F.R. Kschischang, Product Codes, J.G. Proakis (ed), *Wiley encyclopedia of telecommunications*, pp. 2007-2012, vol. 4, Hoboken, NJ, 2003.

[34] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, pp. 498-519, Feb. 2001.

[35] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi and S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "Oceanstore: An architecture for global-scale persistent storage," *9th Int. Conf. on Architectural Support Programm*, pp. 190-201, Cambridge, Massachusetts, 2000.

[36] S. Kudekar, T. Richardson, and R.L. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761-7813, Dec. 2013.

[37] S. Kudekar, M. Mondelli, E. Sasoglu, and R. Urbanke, "Reed-Muller codes achieve capacity on the binary erasure channel under MAP decoding," arXiv:1505.05831v1, 2015.

[38] S. Kumar and H.D. Pfister, "Reed-Muller codes achieve capacity on erasure channels," arXiv:1505.05123v2 , 2015.

[39] A. Lapidoth, “Convolutional codes and finite interleavers for the block erasure channel,” *Mobile Communications Advanced Systems and Components*, Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol. 783, pp. 113-120, May 2005.

[40] S. Lin and D.J. Costello, *Error control coding*, Prentice Hall, 2nd edition, 2004.

[41] F.J. MacWilliams and N.J.A. Sloane, *The theory of error-correcting codes*, North-Holland, 1977.

[42] E. Malkamaki and H. Leib, “Evaluating the performance of convolutional codes over block fading channels,” *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1643-1646, July 1999.

[43] F. Oggier and A. Datta, “Coding techniques for repairability in networked distributed storage systems,” *Foundations and Trends in Communications and Information Theory*, vol. 9, pp. 383-466, 2013.

[44] G.C. Onwubolu and D. Davendra, *Differential evolution: a handbook for global permutation-based combinatorial optimization*, Springer, 2009.

[45] Y. Polyanskiy, H.V. Poor, and S. Verdú, “Channel coding rate in the finite blocklength regime,” *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307-2359, May 2010.

[46] N. Prakash, V. Lalitha, and P.V. Kumar, “Codes with locality for two erasures,” CoRR, vol. abs/1401.2422, 2014. [Online]. Available at arXiv:1401.2422v2, 2014.

[47] R. Pyndiah, “Near-optimum decoding of product codes: Block turbo codes,” *IEEE Trans. Comm.*, vol. 46, no. 8, pp. 1003-1010, Aug. 1998.

[48] D. Rankin and T.A. Gulliver, “Asymptotic performance of product codes,” *IEEE International Conference on Communications*, pp. 431-435, Vancouver, June 1999.

[49] K.V. Rashmi, N.B. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran, “A solution to the network challenges of data recovery in erasure-coded distributed storage systems: A study on the Facebook warehouse cluster,” *Proc. USENIX HotStorage*, June 2013.

[50] S.R. Reddy and J.P. Robinson, “Random Error and Burst Correction by Iterated Codes,” *IEEE Trans. Inf. Theory*, vol. 18, no. 1, pp. 182-185, Jan. 1972.

[51] T.J. Richardson and R.L. Urbanke, *Modern coding theory*, Cambridge University Press, 2008.

[52] E. Rosnes and O. Ytrehus, “Turbo decoding on the binary erasure channel: Finite-length analysis and turbo stopping sets,” *IEEE Trans. Inf. Theory*, vol. 53, no. 11, pp. 4059-4075, Nov. 2007.

[53] E. Rosnes, “Stopping set analysis of iterative row-column decoding of product codes,” *IEEE Trans. Inf. Theory*, vol. 54, no. 4, pp. 1551-1560, April 2008.

[54] A. Sarwate, “Soft decision decoding of Reed-Solomon product codes,” *EECS 229B Final Project Report*, May 2005.

[55] M. Schwartz, P.H. Siegel, and A. Vardy, “On the asymptotic performance of iterative decoders for product codes,” *IEEE International Symposium on Information Theory*, pp. 1758-1762, Sept. 2005.

[56] M. Schwartz and A. Vardy, “On the stopping distance and the stopping redundancy of codes,” *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 922-932, March 2006.

[57] A. Sella and Y. Be’ery, “Convergence analysis of turbo decoding of product codes,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 723-735, Feb. 2001.

[58] N. Sendrier, “Codes correcteurs d’erreurs à haut pouvoir de correction,” Thèse de Doctorat de l’Université Paris 6, in French, Dec. 1991.

[59] N.J.A Sloane, The On-Line Encyclopedia of Integer Sequences. See sequence A000041 at oeis.org/A000041.

[60] R.P. Stanley, *Enumerative combinatorics*, Cambridge Univ. Press, vol. 1, 2nd edition, 2012.

[61] R. Storn and K. Price, “Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.

[62] R.M. Tanner, “A recursive approach to low complexity codes,” *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533-547, Sept. 1981.

[63] J.-P. Tillich and G. Zémor, “Optimal cycle codes constructed from Ramanujan graphs,” *SIAM J. Discrete Mathematics*, vol. 10, no. 3, pp. 447-459, 1997.

[64] L.M.G.M. Tolhuizen, “More results on the weight enumerator of product codes,” *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2537-2577, Sept. 2002.

[65] D.N.C. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, Cambridge University Press, 2005.

[66] W.M.C.J. Van Overveld, “Multiple-burst error-correcting cyclic product codes (Corresp.),” *IEEE Trans. Inf. Theory*, vol. 33, no. 6, pp. 919-923, Nov. 1987.

[67] D.P. Varodayan, “Investigation of the Elias product code construction for the binary erasure channel”, B.A.S. Thesis, University of Toronto, Dec. 2002.

[68] S. Wainberg, “Error-erasure decoding of product codes (Corresp.),” *IEEE Trans. on Inf. Theory*, vol. 18, no. 6, pp. 821-823, Nov. 1972.

[69] V.K. Wei, “Generalized Hamming weights for linear codes,” *IEEE Trans. Inf. Theory*, vol. 37, no. 5, pp. 1412-1418, Sept. 1991.

[70] L.-J. Weng and G. Sollman, “Variable redundancy product codes,” *IEEE Trans. on Comm. Technology*, vol. 15, no. 6, pp. 835-838, Dec. 1967.

[71] S.B. Wicker and V.K. Bhargava, eds., *Reed-Solomon Codes and their Applications*. New York: IEEE Press, 1994.

[72] J.K. Wolf, “On codes derivable from the tensor product of check matrices,” *IEEE Trans. Inf. Theory*, vol. 11, no. 2, pp. 281-284, April 1965.