Generalized and Doubly Generalized LDPC Codes With Random Component Codes for the Binary Erasure Channel

Enrico Paolini, *Member, IEEE*, Marc P. C. Fossorier, *Fellow, IEEE*, and Marco Chiani, *Senior Member, IEEE*

***Abstract*—In this paper, a method for the asymptotic analysis of generalized low-density parity-check (GLDPC) codes and doubly generalized low-density parity-check (D-GLDPC) codes over the binary erasure channel (BEC), based on extrinsic information transfer (EXIT) chart, is described. This method overcomes the problem consisting of the impossibility to evaluate the EXIT function for the check or variable component codes, in situations where the information functions or split information functions for component codes are unknown. According to the proposed technique, GLDPC codes and D-GLDPC codes where the generalized check and variable component codes are *random* codes with minimum distance at least 2, are considered. A technique is then developed which finds the EXIT chart for the overall GLDPC or D-GLDPC code, by evaluating the expected EXIT function for each check and variable component code. This technique is finally combined with the differential evolution algorithm in order to generate some good GLDPC and D-GLDPC edge distributions. Numerical results of long, random codes, are presented which confirm the effectiveness of the proposed approach. They also reveal that D-GLDPC codes can outperform standard LDPC codes and GLDPC codes in terms of both waterfall performance and error floor.**

***Index Terms*—Binary erasure channel, channel coding, EXIT chart, information functions, iterative decoding, low-density parity-check (LDPC) codes, split information functions.**

I. Introduction

LOW-DENSITY parity-check (LDPC) codes [1] have been shown to exhibit excellent asymptotic performance over a wide range of channels, under iterative decoding [2], [3]. It has been proved that irregular LDPC codes are able to asymptotically achieve the binary erasure channel (BEC) capacity for any code rate [4], [5]: this means that, for any code rate R and for any small $\epsilon > 0$, it is possible to design an edge degree distribution (λ, ρ) such that $\int_0^1 \rho(x)\mathrm{d}x / \int_0^1 \lambda(x)\mathrm{d}x = 1 - R$ and whose asymptotic threshold is $q^* = (1-\epsilon)(1-R)$. Examples of capacity achieving (sequences of) degree distributions are the heavy-tail Poisson sequence [4] and the binomial sequence [6].

It is well known that this very good asymptotic performance in terms of decoding threshold does not necessarily correspond to a satisfying finite length performance. In fact, finite length LDPC codes with good asymptotic threshold, though typically characterized by good waterfall performance, are usually affected by high error floors [7]–[9]. This phenomenon has been partly addressed in [10], where it is proved that all the so far known capacity approaching LDPC degree distributions, all characterized by $\lambda'(0)\rho'(1) > 1$, are associated with finite length LDPC codes whose minimum distance is a sublinear (more precisely, logarithmic) function of the codeword length N. When considering transmission on the BEC, the low weight codewords induce small stopping sets [11], thus resulting in high error floors.

The (so far not overcome) inability in generating LDPC codes with threshold close to capacity and good minimum distance properties as well, is one of the main motivations for investigating more powerful (and complex) coding schemes. Such examples are generalized LDPC (GLDPC) codes and doubly generalized LDPC (D-GLDPC) codes. In GLDPC codes, generic linear block codes are used as check nodes (CNs) in addition to the traditional single parity-check (SPC) codes. First introduced in [12], GLDPC codes have been more recently investigated, for instance, in [13]–[23]. Recently introduced in [24], [25] (see also the previous work [26]), D-GLDPC codes represent a wider class of codes than GLDPC codes. The generalization consists of using generic linear block codes as variable nodes (VNs) in addition to the traditional repetition codes. Linear block codes used as check or variable nodes are called *component codes* of the D-GLDPC code. The CNs represented by component codes which are not SPC codes are called *generalized check nodes*, and their associated codes *generalized check component codes*. Analogously, the VNs represented by component codes which are not repetition codes are called *generalized variable nodes*, and their associated codes *generalized variable component codes*. The ensemble of all the CNs is referred to as the check node set, and the ensemble of all the VNs as the variable node set. In this paper, only check and variable (linear) binary component codes are considered, so that the overall GLDPC or D-GLDPC code is a binary code.

It is worthwhile pointing out a connection between D-GLDPC codes and the class of expander codes constructed on bipartite graphs investigated, for instance, in [27], [28] and

Manuscript received May 08, 2007; revised March 02, 2009. Current version published March 17, 2010. This research was supported by the NSF under Grant CCF-0515154, by ESA/ESOC and by the European Community under Seventh Framework Program grant agreement ICT OPTIMIX n.INFSO-ICT-214625. The material in this paper was presented in part at the Forty-Fourth Allerton Conference on Communications, Control & Computing, Monticello, IL, September 2006 and in part at the International Symposium on Information Theory and its Applications (ISITA), Seoul, Korea, November 2006.

E. Paolini and M. Chiani are with DEIS/WiLAB, University of Bologna, 47521 Cesena (FC), Italy (e-mail: e.paolini@unibo.it, marco.chiani@unibo.it).

Marc Fossorier is with ETIS ENSEA/UCP/CNRS/UMR-8051, 95014 Cergy Pontoise, France (e-mail: mfossorier@ieee.org).

Communicated by T. J. Richardson, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2010.2040938

referred to in this latter work as *bipartite graph codes*. Considering the code construction described in [28, Section II.A], each node in the bipartite graph is associated with a binary linear code, and each edge in the bipartite graph is associated with an encoded bit. A binary word is a valid codeword for the bipartite graph code if and only if each node in the graph recognizes a valid local codeword. (Note that the regular construction described in [27], [28] can be easily extended to irregular constructions.) A D-GLDPC code whose VNs are all represented in systematic form can be interpreted as a punctured bipartite graph code, where the punctured bits are those associated with the local parity bits of each VN.

This interpretation is coherent with the representation of these codes as GLDPC codes. A bipartite graph code can be represented as a GLDPC code where all VNs have a degree 2 (in the same way as the *code-to-code graph* illustrated in [21]). Moreover, a D-GLDPC code can be represented as a punctured GLDPC code, provided all the VNs of the D-GLDPC code are represented in systematic form [29].[1] If we now consider a D-GLDPC code with VNs in systematic form, we can represent it either as a punctured GLDPC code or as a punctured bipartite graph code. If we now represent this latter code as a GLDPC code we obtain again the same punctured GLDPC code. This interpretation of a D-GLDPC codes as a punctured bipartite graph code is however limited to the case where all D-GLDPC code VNs are represented in systematic form.

The asymptotic threshold analysis of random GLDPC codes and random D-GLDPC codes can be in principle performed through extrinsic information transfer (EXIT) charts [30]–[32]. The success of this approach is bound to the knowledge of the EXIT function for each check and variable component code. In [31, Th. 2] it is proved that, if the communication channel is a BEC, then the EXIT function of a linear block code, under *maximum a posteriori* probability (MAP) decoding, can be related to the code *information functions* (a concept first introduced in [33]), and that the EXIT function of a linear block code with split encoder, under MAP decoding, to the code *split information functions*. This relationship between EXIT function and (split) information functions is very useful for the threshold analysis over the BEC of GLDPC and D-GLDPC codes constructed with component codes whose (split) information functions are known. A major problem is that, for a wide range of linear block codes, including most binary double error-correcting and more powerful Bose–Chaudhuri–Hocquenghen (BCH) codes, these parameters are still unknown. In fact, no closed-form expression is available as a function of the code dimension k and code length n, and a direct computation of these parameters is often unfeasible, due to the huge computation time required, even for small codeword lengths. This is the case, for instance, of the split information function computations for the $(31, 10)$ dual code of a narrowsense binary $(31, 21)$ BCH code.

In this paper, a solution is proposed for the asymptotic analysis of GLDPC and D-GLDPC codes, which allows to overcome the impossibility to evaluate the EXIT function for the check or variable component codes when the above-mentioned code parameters become too large. The proposed method consists of considering *random* check and variable generalized component codes belonging to a certain expurgated ensemble, instead of specific check and variable generalized component codes (like Hamming codes, BCH codes, etc., or their dual codes). This expurgated ensemble is the ensemble of all the binary linear block codes with given codeword and information block lengths and whose minimum distance satisfies $d_{\min} \geq 2$. A technique is then developed to exactly evaluate the expected information function for each check component code and the expected split information function for each variable component code over the expurgated ensemble. This allows to evaluate the expected EXIT function for each check component code or variable component code, assuming transmission over a BEC, and therefore the expected EXIT function for the overall CN set or VN set.

The developed analytical tool is then exploited to design capacity approaching GLDPC and D-GLDPC distributions. Simulation results obtained on random, long codes, reveal that capacity approaching D-GLDPC codes can be characterized by a better threshold and a lower error floor than capacity approaching LDPC and GLDPC codes, at the cost of increased decoding complexity. Moreover, by imposing constraints on the fraction of edges toward the generalized CNs, the error floor of D-GLDPC codes can be further lowered, while preserving a good waterfall performance.

The paper is organized as follows. In Section II, the concept of GLDPC code and D-GLDPC code is presented, while in Section III the relationship between the EXIT function of check and variable component codes, and (split) information functions, is reviewed for the BEC. In the same section, the EXIT function of a generalized CN, assuming bounded distance decoding instead of MAP decoding, is investigated for the BEC. Section IV is devoted to the derivation of the random component code constraints and to the definition of the expurgated ensemble of check and variable component codes, which guarantees a correct application of the EXIT chart analysis. In Sections V and VI, the evaluations of the expected information function for a random check component code and of the expected split information function for a random variable component code are developed, respectively. Numerical results involving threshold analysis, distribution optimization and finite-length performance analysis, for both GLDPC and D-GLDPC codes, are presented in Section VII. Finally, concluding remarks are given in Section VIII.

II. GLDPC Codes and D-GLDPC Codes

A traditional LDPC code of length N and dimension K is usually graphically represented by means of a bipartite graph, known as a Tanner graph [12], characterized by N VNs and a number $M \geq N - K$ of CNs. Each edge in the graph can only connect a VN to a CN. According to this representation, the VNs have a one-to-one correspondence with the encoded bits of the codeword, and each CN represents a parity-check equation involving a certain number of encoded bits. The degree of a node is defined as the number of edges connected to the node. Thus, the degree of a VN is the number of parity constraints the corresponding encoded bit is involved in, and the degree of a CN

[1]Note that the D-GLDPC and GLDPC representations are equivalent in the sense that they share the same set of codewords, but the equivalence does not hold for the iterative decoders.

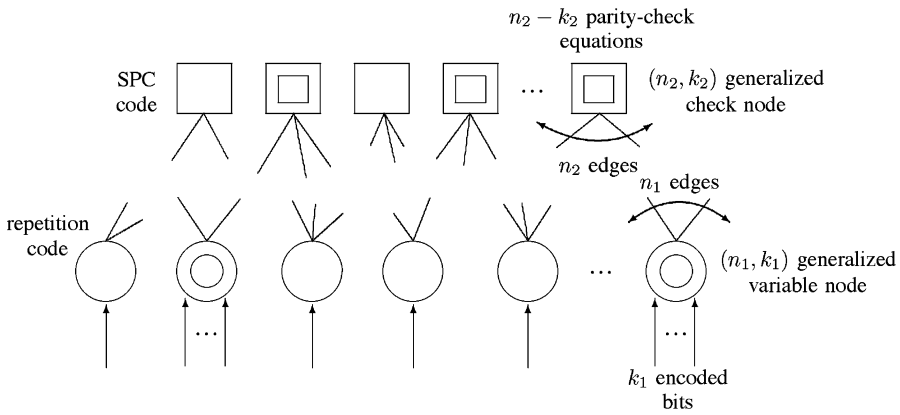

Fig. 1. Structure of a D-GLDPC code.

is the number of bits involved in the corresponding parity-check equation.

A degree-n CN of a standard LDPC code can be interpreted as a length-n SPC code, i.e., as a $(n, n-1)$ linear block code. Analogously, a degree-n VN can be interpreted as a length-n repetition code, i.e., as a $(n, 1)$ linear block code, where the information bit corresponds to the bit received from the communication channel. A first step toward the generalization of LDPC codes consists of letting some of (or eventually all) the CNs, be generic (n, k) linear block codes: the corresponding code structure is known as a GLDPC code. An (n, k) generalized CN is connected to n VNs, and corresponds to $n - k$ parity-check equations. Then, for GLDPC codes, the number of parity check equations is no longer equal to the number of CNs.

The generalized CNs are characterized by a higher error or erasure correction capability than SPC codes, and they can be favorable from the viewpoint of the code minimum distance [13], [22]. The drawback of using generalized CNs is an overall code rate loss, which might not be compensated by their higher correction capability [18]. This makes GLDPC codes with a uniform CN structure (i.e., composed only of generalized nodes, e.g., only of $(7, 4)$ Hamming codes) quite poor in terms of decoding threshold. This poor threshold, which was evaluated in [18] for the BEC assuming bounded distance decoding at the CNs, does not improve to competitive values even if MAP decoding is performed at the CNs, as it will be shown in Section VII.

The second generalization step consists of introducing VNs different from repetition codes. The corresponding code structure is known as D-GLDPC code [24], [25], [34], and is represented in Fig. 1. An (n, k) generalized VN is connected to n CNs, and receives its k information bits from the communication channel. Thus, k of the N encoded bits for the overall D-GLDPC code are received by the (n, k) generalized VN, and interpreted by the VN as its k information bits.

The above mentioned rate loss introduced by generalized CNs makes GLDPC codes an attractive solution only for low code rate coding schemes. On the other hand, the introduction of generalized variable component codes enables a larger flexibility in terms of code rate, due to the possibility to employ VNs with a higher local code rate than repetition VNs for the same node degree. For example, consider a GLDPC ensemble where all the CNs are of the same type and all the VNs have the same degree. If $(7, 4)$ Hamming codes are chosen as check component codes, then the only possible choice for the degree of the VNs is 2, resulting in an overall code rate $R = 1/7$. Additionally, letting the VNs be linear block codes other than repetition codes enables to achieve a much wider range of code rates. A rate $R = 1/2$ is achieved if all the VNs have a local code rate equal to $6/7$, i.e., if they are SPC codes of length 7. A high rate $R > 1/2$ is achieved if the local code rate of the VNs is larger than $6/7$ and a lower rate $R < 1/2$ is achieved otherwise.

The iterative decoding algorithm for GLDPC and D-GLDPC codes over the BEC is a generalization of the iterative decoder for LDPC codes presented in [4]. Suppose that MAP decoding is performed at each check and variable component code. In the first half of each decoding iteration (*horizontal step*), a generic (n, k) CN receives n messages from its neighboring VNs: Some of them are known messages (i.e., "0" or "1" messages, with infinite reliability), others are erasure messages (i.e., "?" messages). MAP decoding is then performed at the CN in order to recover its unknown encoded bits. After the CN has completed its decoding procedure, a known message is sent toward the VNs for each known encoded bit, along the corresponding edge, while an erasure message is sent for each encoded bit which remains unknown.

In the second half of each decoding iteration (*vertical step*), a generic (n, k) VN receives n messages from its neighboring CNs: Again, some of them are known messages, others are erasure messages. The decoding of an (n, k) VN is analogous to that of a CN, with the difference that, at each iteration, some of the information bits (observed from the communication channel) might be known as well as some of the encoded bits. In order to exploit the partial knowledge of the information bits, MAP decoding is performed on the extended generator matrix $[\boldsymbol{G} \,|\, \boldsymbol{I}_k]$, where $\boldsymbol{G}$ is the $(k \times n)$ generator matrix chosen to represent the VN and $\boldsymbol{I}_k$ is the $(k \times k)$ identity matrix. After MAP decoding has been performed, some of the previously unknown information and encoded bits for the VN might be recovered. Known messages are then sent to the CNs associated with known encoded bits, while erasure messages are sent for the encoded bits which remain unknown.

The algorithm is stopped as soon as there are no longer unknown encoded bits for the overall (N, K) code (in this case

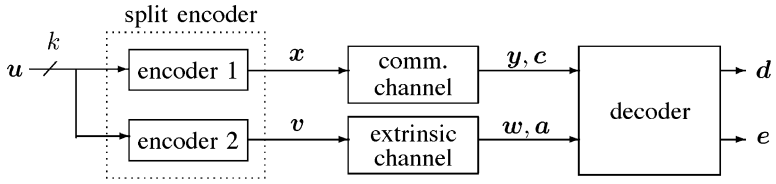

Fig. 2. General decoding model for the variable and check nodes of a D-GLDPC code.

decoding is successful), or when unknown encoded bits for the overall (N, K) code still exist, but either a maximum number of iterations has been reached, or in the last iteration no encoded bits were recovered (in these cases, a decoding failure is declared).

With respect to iterative decoding of LDPC codes over the BEC, during iterative decoding of D-GLDPC codes over the BEC, each generalized VN or CN node may not be updated only once if MAP decoding is used at the nodes of the graph. In fact, during a certain iteration, MAP decoding performed locally at a generalized VN or CN may recover some of the unknown local code bits but not all of them, so that at the next iteration the node needs a further MAP decoding processing.[2] On the other hand, similarly to iterative decoding of LDPC codes over the BEC, also during iterative decoding of D-GLDPC codes over the BEC each edge of the bipartite graph is used only once.

III. Exit Functions for Generalized Variable and Check Nodes Over the BEC

In [31, Fig. 3], a general decoding model is described, which can be effectively adopted to express the EXIT function over the BEC for the generalized CNs and generalized VNs of a D-GLDPC code. This general decoding model is depicted in Fig. 2. The encoder of a linear block code with dimension k is split into two linear encoders, namely *encoder 1* and *encoder 2*, with generator matrices $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$. The encoder generates a codeword $(\boldsymbol{x}, \boldsymbol{v})$, with $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{G}_1$ and $\boldsymbol{v} = \boldsymbol{u}\boldsymbol{G}_2$, and codeword length $|\boldsymbol{x}|+|\boldsymbol{v}|$, where $|\cdot|$ denotes the length of a vector. For each information word $\boldsymbol{u}$, the encoded bits $\boldsymbol{x}$ are transmitted over a *communication channel*, resulting in $\boldsymbol{y}$, while the encoded bits $\boldsymbol{v}$ are transmitted over an *extrinsic channel*, resulting in $\boldsymbol{w}$. Both the likelihood ratios $\boldsymbol{c}$ and $\boldsymbol{a}$, relative to $\boldsymbol{y}$ and $\boldsymbol{w}$, respectively, are exploited by the *a posteriori* probability (APP) decoder in order to compute the likelihood ratios $\boldsymbol{d}$ for the encoded bits, and the extrinsic likelihood ratios $\boldsymbol{e}$. In the following, capital letters denote random variables and lower case letters realizations of such random variables.

The EXIT function of the linear code in Fig. 2, assuming that the encoders 1 and 2 have no idle bits[3], that MAP decoding is performed, that the communication channel is a BEC with erasure probability q and that the extrinsic channel is a BEC with erasure probability p, has been shown in [31, eq. 36] to be expressed by

$$\begin{aligned} I_E(p,q) &\triangleq \frac{1}{|\boldsymbol{v}|}\sum_{i=1}^{|\boldsymbol{v}|} I(V_i; E_i) \\ &= \frac{1}{|\boldsymbol{v}|}\sum_{i=1}^{|\boldsymbol{v}|} I(V_i; \boldsymbol{Y}, \boldsymbol{W}_{[i]}) \\ &= 1 - \frac{1}{|\boldsymbol{v}|}\sum_{h=0}^{|\boldsymbol{x}|}(1-q)^h q^{|\boldsymbol{x}|-h}\sum_{g=1}^{|\boldsymbol{v}|}(1-p)^{g-1}p^{|\boldsymbol{v}|-g} \\ &\quad \times [g\,\tilde{e}_{g,h} - (|\boldsymbol{v}| - g + 1)\tilde{e}_{g-1,h}]. \end{aligned} \tag{1}$$

In (1), V_i is the ith bit of the encoded word $\boldsymbol{V}$ (that is a Bernoulli random variable with equiprobable values), E_i is the extrinsic log-likelihood ratio associated with the ith encoded bit $v_i \in \boldsymbol{v}$, $\boldsymbol{Y}$ is the random word outcoming from the communication channel, $\boldsymbol{W}_{[i]}$ is the random word outcoming from the extrinsic channel except its element W_i, $I(\cdot)$ denotes the mutual information, and $\tilde{e}_{g,h}$ is the (g,h)th *unnormalized split information function*. This parameter is defined as the summation of the dimensions of all the possible codes obtained by considering g positions among the encoded bits $\boldsymbol{v}$ and h positions among the encoded bits $\boldsymbol{x}$. It can be computed by performing the summation of the ranks of the $\binom{|\boldsymbol{v}|}{g} \cdot \binom{|\boldsymbol{x}|}{h}$ submatrices obtained by selecting g columns in $\boldsymbol{G}_2$ and h columns in $\boldsymbol{G}_1$. Note that the equality $I(V_i; E_i) = I(V_i; \boldsymbol{Y}, \boldsymbol{W}_{[i]})$, proved in [31, Proposition 1], remains valid (under MAP decoding) over channels other than the BEC.

We refer to this decoding model in order to describe and analyze each generalized CN and each generalized VN of a D-GLDPC code. Within the context of GLDPC and D-GLDPC codes, the communication channel is the channel over which the encoded bits of the overall code are transmitted, while the extrinsic channel represents a model for the channel over which the messages are exchanged between variable and check nodes, during the iterative decoding process. Coherently with the the description of the decoding algorithm presented in the previous section, if the communication channel is a BEC, then also the extrinsic channel can be modeled as a BEC.

A. EXIT Function for the Variable Nodes Over the BEC

The generic VN (either repetition or generalized), representing an (n, k) linear block code, receives its k information bits from the communication channel, and interfaces with the extrinsic channel through its n encoded bits. For this reason, for a VN the encoder 1 is represented by the identity mapping $\boldsymbol{x} = \boldsymbol{u}$ (i.e., $\boldsymbol{G}_1 \sim \boldsymbol{I}_k$) and the encoder 2 performs the linear

[2]Each VN or CN is updated only once if bounded distance decoding is used at the node instead of MAP decoding. Note that, for repetition VNs and SPC CNs, MAP and bounded distance decoding algorithms are equivalent.

[3]The expression "linear block code with no idle bits" is used to indicate that the generator matrix of the linear block code has no all-zero columns (the expression "no idle components" is used in [31]). A linear block code with no idle bits is often referred to as a "proper code".

mapping $\boldsymbol{v} = \boldsymbol{G}\boldsymbol{u}$ (i.e., $\boldsymbol{G}_2 = \boldsymbol{G}$), where $\boldsymbol{G}$ is the generator matrix chosen to represent the (n,k) linear block code. In this case it results $|\boldsymbol{x}| = k$ and $|\boldsymbol{v}| = n$. Thus, the (n,k) VN may be interpreted as a $(n+k,k)$ code whose generator matrix is in the form $[\boldsymbol{G}\,|\,\boldsymbol{I}_k]$, and its EXIT function over the BEC is given by (1), with the encoder 1 being the identity mapping $\boldsymbol{x} = \boldsymbol{u}$ and the encoder 2 being the linear mapping $\boldsymbol{v} = \boldsymbol{G}\boldsymbol{u}$. This EXIT function can be equivalently expressed by:

$$I_E(p,q) = 1 - \frac{1}{n}\sum_{t=0}^{n-1}\sum_{z=0}^{k} p^t\,(1-p)^{n-t-1}\,q^z\,(1-q)^{k-z} \times[(n-t)\tilde{e}_{n-t,k-z} - (t+1)\tilde{e}_{n-t-1,k-z}] \quad (2)$$

which can be easily obtained from (1) by performing the substitutions $t = n - g$ and $z = k - h$. Expressions (1) and (2) are valid under the hypothesis of MAP erasure decoding. If applied to a $(n,1)$ repetition code, (2) leads to $I_E(p,q) = 1 - qp^{n-1}$, i.e., to the well known expression of the EXIT function for a degree-n VN of an LDPC code over the BEC.

We observe that the split information functions are not univocal for a given (n,k) generalized VN and depend on the specific representation chosen for its generator matrix $\boldsymbol{G}$. This can be justified as follows. Different generator matrices correspond to different mappings of vectors $\boldsymbol{u}$'s to vectors $\boldsymbol{v}$'s. Hence, for a given information vector $\boldsymbol{u}$, a generator matrix $\boldsymbol{G}'$ leads to a codeword $[\boldsymbol{v}'|\boldsymbol{u}]$ for the $(n+k,k)$ code with split encoder, while a generator matrix $\boldsymbol{G}''$ leads to a different codeword $[\boldsymbol{v}''|\boldsymbol{u}]$, thus generating a different code book. As a consequence, the EXIT function for a (n,k) linear block code when used as a VN of a D-GLDPC code, depends on the generator matrix representation chosen for the code. This fact does not hold for repetition codes (i.e., for the traditional VNs of LDPC and GLDPC codes), for which only one code representation is possible. Then, an important difference between VNs represented by repetition codes and generalized (n,k) VNs of a D-GLDPC code (with $k > 1$) is that, in the latter case, different representations of the generator matrix $\boldsymbol{G}$ are possible. These different VN representations correspond to *different performances* of the overall D-GLDPC codes. Therefore, two generalized VNs associated with different representations of the same linear block code shall be considered to belong to different variable component code types. The code representation for the generalized VNs becomes a degree of freedom for the code design.

B. EXIT Function for the Check Nodes Over the BEC

For a CN (either SPC or generalized), no communication channel is present. Moreover, any CN representing an (n,k) linear block code interfaces with the extrinsic channel through its n encoded bits. Then, the encoder 1 is not present, while the encoder 2 performs the linear mapping $\boldsymbol{v} = \boldsymbol{G}\boldsymbol{u}$, where $\boldsymbol{G}$ is one of the several possible generator matrix representations for the (n,k) linear block code. It follows that $|\boldsymbol{v}| = n$. This model is the same as that proposed in [31, Sec. VII-A].

The EXIT function of a generic (n,k) CN of a D-GLDPC code on the BEC can be obtained by letting $q \to 1$ in (2) (no communication channel is present). The obtained expression, equivalent to [31, eq. 40], is

$$\begin{aligned} I_E(p) &\triangleq \frac{1}{|\boldsymbol{v}|}\sum_{i=1}^{|\boldsymbol{v}|} I(V_i;E_i) \\ &= \frac{1}{|\boldsymbol{v}|}\sum_{i=1}^{|\boldsymbol{v}|} I\left(V_i;\boldsymbol{W}_{[i]}\right) \\ &= 1 - \frac{1}{n}\sum_{t=0}^{n-1} p^t(1-p)^{n-t-1} \\ &\quad \times\left[(n-t)\,\tilde{e}_{n-t} - (t+1)\tilde{e}_{n-t-1}\right] \end{aligned} \quad (3)$$

where, for $g = 0,\ldots,n$, $\tilde{e}_g$ is the gth (un-normalized) information function [33]. It is defined as the summation of the dimensions of all the possible codes obtained by considering g positions in the code block $\boldsymbol{v}$ of length n. This parameter can be computed by performing the summation of the ranks of the $\binom{n}{g}$ submatrices obtained by selecting g columns in $\boldsymbol{G}$.

As (2), (3) assumes that erasures are corrected at the CN according to MAP decoding. If applied to a $(n,n-1)$ SPC code, (3) leads to the expression of the EXIT function on the BEC for a degree-n CN of an LDPC code, i.e., $I_E(p) = (1-p)^{n-1}$.

Since the code book of a linear block code is independent of the choice of its generator matrix, different code representations have the same information function. Thus, different code representations for a generalized CN lead to the same EXIT function. This means that, differently from what happens for the generalized VNs, the performance of a GLDPC or D-GLDPC code is independent of the specific representation of its generalized CNs.

C. Bounded-Distance EXIT Functions

Decoding algorithms less powerful than MAP decoding, but having a reduced complexity, may be used at the generalized variable and check nodes. In these cases, different expressions of the EXIT function must be considered. For example, consider a generalized (n,k) CN, and assume that erasure recovery is performed according to the following bounded-distance decoding strategy, referred to as *d-bounded-distance decoding*: "if the number of received erasures from the extrinsic channel is less than or equal to d, execute MAP decoding, otherwise declare a decoding failure".

Theorem 1: If the extrinsic channel is a BEC with erasure probability p, then the EXIT function for a generalized (n,k) CN without idle bits, under d-bounded-distance decoding, is given by

$$I_E(p) = 1 - \frac{1}{n}\sum_{t=0}^{d-1}(1-p)^{n-t-1}p^t[(n-t) \times\tilde{e}_{n-t} - (t+1)\tilde{e}_{n-t-1}] - \sum_{t=d}^{n-1}(1-p)^{n-t-1}p^t\binom{n-1}{t}. \quad (4)$$

Proof: See Appendix I. □

A nontrivial consequence of Theorem 1 is that, over the BEC, the equality $I(V_i;E_i) = I(V_i;\boldsymbol{W}_{[i]})$ remains valid if d-bounded-distance decoding is employed instead of MAP decoding. In fact, it is readily shown that (4) can be obtained both from $I_E = I(V_i;\boldsymbol{W}_{[i]})$ and from $I_E = I(V_i;E_i)$.

Example 1: In [18, Table 2], some thresholds on the BEC are presented for GLDPC codes with a uniform CN structure, composed of narrow-sense binary BCH codes. These thresholds have been evaluated through density evolution, assuming d-bounded-distance decoding at the CNs, with $d = d_{\min} - 1$. The same thresholds can also be obtained through an EXIT chart approach exploiting (4) for the BCH codes, with $d = d_{\min} - 1$.

IV. Random Component Code Hypothesis

A. Definitions

Consider a D-GLDPC code with n'_v variable component code types and n'_c check component code types. Any type-t variable node ($t \in \{1,\ldots,n'_v\}$) has EXIT function over the BEC $I^{(t)}_{E,V}(p,q)$ (corresponding to a specific code representation), and is assumed to have no idle components. Analogously, any type-t CN ($t \in \{1,\ldots,n'_c\}$) has EXIT function $I^{(t)}_{E,C}(p)$, and is assumed to have no idle components. Variable and check nodes are assumed to be *randomly connected* through an edge interleaver. The fraction of edges incident on the variable nodes of type t is denoted by λ'_t for $t \in \{1,\ldots,n'_v\}$), and the fraction of edges incident on the CNs of type t by ρ'_t for $t \in \{1,\ldots,n'_c\}$). Then, the EXIT functions of the VN set and CN set are given by

$$I_{E,V}(p,q) = \sum_{t=1}^{n'_v} \lambda'_t\, I^{(t)}_{E,V}(p,q) \tag{5}$$

and

$$I_{E,C}(p) = \sum_{t=1}^{n'_c} \rho'_t\, I^{(t)}_{E,C}(p) \tag{6}$$

respectively. The relationships (5) and (6) can be obtained by reasoning in the same way as in [31, Example 7] for the EXIT functions of the variable and check node sets of an irregular LDPC code.

Definition 1 (Independent Set and Independent Column): Given a $(k \times n)$ binary matrix of rank r, a set of t columns is called an *independent set*[4] when removing these t columns from the matrix leads to a $(k \times (n-t))$ matrix with rank $r - \Delta r < r$, for some $0 < \Delta r \le t$. The number t is the size of the independent set. An independent set of size $t = 1$ is also called an *independent column*. An independent column is linearly independent of all the other columns of the matrix.

If j columns form an independent set for a $(k \times n)$ binary matrix $\boldsymbol{M}$ of rank r, then they form an independent set for any $(k \times n)$ matrix obtained by summing to any row of $\boldsymbol{M}$ other rows of $\boldsymbol{M}$. Moreover, removing these columns from any such matrix leads to the same rank reduction Δr. This is because such operations performed on $\boldsymbol{M}$ cannot alter the rank of $\boldsymbol{M}$ or of submatrices composed of columns of $\boldsymbol{M}$. Hence, if j columns form an independent set for a generator matrix of a linear block code, then they form an independent set for any other generator matrix of the same code. Moreover, removing these columns from any representation of the generator matrix leads to the same rank reduction.

Definition 2 (Expurgated Ensemble of $(k \times n)$ Matrices): Let $\mathcal{M}^{(n,k)}$ denote the ensemble of all the $(k \times n)$ binary matrices. Moreover, for $k < n$ let $\mathcal{G}^{(n,k)}$ denote the ensemble of all the $(k \times n)$ binary matrices of rank k, that is the ensemble of all the $(k \times n)$ binary matrices representing binary linear block codes of length n and dimension k. We define expurgated ensemble of $(k \times n)$ generator matrices the ensemble $\mathcal{G}^{(n,k)}_* \subseteq \mathcal{G}^{(n,k)} \subseteq \mathcal{M}^{(n,k)}$ of all the $(k \times n)$ binary matrices with rank k, without all-zero columns and without independent columns.

Definition 3 (Random Component Code Hypothesis): A D-GLDPC code ensemble is said to fulfill the random component code hypothesis when the two following conditions hold: 1) any variable component code is a random linear block code whose generator matrix is randomly chosen, with uniform probability, from the expurgated ensemble $\mathcal{G}^{(n,k)}_*$, where n and k are the length and the dimension of the variable component code, respectively; 2) any check component code is a random linear block code whose generator matrix is randomly chosen, with uniform probability, from the expurgated ensemble $\mathcal{G}^{(n,k)}_*$, where n and k are the length and the dimension of the check component code, respectively.

Let us consider a D-GLDPC ensemble fulfilling the random component code hypothesis. Assume the VN set is partitioned into a number n_v of subsets, where the ith subset ($i \in \{1,\ldots,n_v\}$) is the ensemble of all the VNs whose generator matrix is randomly drawn from $\mathcal{G}^{(n_i,k_i)}_*$, for the same n_i and k_i. Similarly, assume the CN set is partitioned into a number n_c of subsets, where the ith subset ($i \in \{1,\ldots,n_c\}$) is the ensemble of all the CNs whose generator matrix is randomly drawn from $\mathcal{G}^{(n_i,k_i)}_*$, for the same n_i and k_i. Furthermore, let us denote by λ_i the fraction of edges incident on VNs belonging to the ith such subset of the VN set, and by ρ_i the fraction of edges incident on CNs belonging to the ith such subset of the CN set. Let $\mathbb{E}[I_{E,V}(p,q)]$ and $\mathbb{E}[I_{E,C}(p)]$ be the expected EXIT functions of the variable and check node set, respectively. From (5) and (6) we have

$$\begin{aligned}\mathbb{E}[I_{E,V}(p,q)] &= \mathbb{E}\left[\sum_{t=1}^{n'_v} \lambda'_t\, I^{(t)}_{E,V}(p,q)\right] \\ &= \sum_{i=1}^{n_v} \lambda_i\, \mathbb{E}_{\mathcal{G}^{(n_i,k_i)}_*}\left[I^{(i)}_{E,V}(p,q)\right] \end{aligned} \tag{7}$$

$$\begin{aligned}\mathbb{E}[I_{E,C}(p)] &= \mathbb{E}\left[\sum_{i=t}^{n'_c} \rho'_t\, I^{(t)}_{E,C}(p)\right] \\ &= \sum_{i=1}^{n_c} \rho_i\, \mathbb{E}_{\mathcal{G}^{(n_i,k_i)}_*}\left[I^{(i)}_{E,C}(p)\right] \end{aligned} \tag{8}$$

[4]We prefer to use the expression "independent set" even if there may exist columns in the set that are not linearly independent of the other columns in the matrix. As commented by one of the Reviewers, the expression "independent set" is intended here as "not fully dependent set".

In (7), $\mathbb{E}_{\mathcal{G}_*^{(n_i,k_i)}}[I_{E,V}^{(i)}(p,q)](i \in \{1,\ldots,n_v\})$ is the expected EXIT function over $\mathcal{G}_*^{(n_i,k_i)}$ for a VN whose generator matrix is randomly drawn from $\mathcal{G}_*^{(n_i,k_i)}$. Note that, if $k_i = 1$ (repetition code), expectation may be omitted. In (8), $\mathbb{E}_{\mathcal{G}_*^{(n_i,k_i)}}[I_{E,C}^{(i)}(p)]$ $(i \in \{1,\ldots,n_c\})$ is the expected EXIT function over $\mathcal{G}_*^{(n_i,k_i)}$ for a CN whose generator matrix is randomly drawn from $\mathcal{G}_*^{(n_i,k_i)}$.

The reason for an expurgated ensemble $\mathcal{G}_*^{(n,k)}$ of $(k \times n)$ generator matrices, instead of either the ensemble $\mathcal{G}^{(n,k)}$ of all the possible $(k \times n)$ binary matrices with rank k or the ensemble $\mathcal{M}^{(n,k)}$ of all the $(k \times n)$ binary matrices, is to ensure a correct application of the EXIT chart analysis, as explained next.

Theorem 2: For a D-GLDPC ensemble fulfilling the random component code hypothesis, the following relationships hold:

$$\lim_{p\to 0} \mathbb{E}[I_{E,V}(p,q)] = 1 \quad \forall q \in (0,1) \tag{9}$$

$$\lim_{p\to 0} \mathbb{E}[I_{E,C}(p)] = 1 \tag{10}$$

$$\lim_{p\to 1} \mathbb{E}[I_{E,C}(p)] = 0. \tag{11}$$

Proof: Since we have $\sum_{i=1}^{n_v} \lambda_i = 1$ and $\sum_{i=1}^{n_c} \rho_i = 1$, it follows from (7) and (8) that (9) and (10) are satisfied if the following properties hold:

$$\lim_{p\to 0} \mathbb{E}_{\mathcal{G}_*^{(n_i,k_i)}}[I_{E,V}^{(i)}(p,q)] = 1 \quad \forall q \in (0,1) \quad \forall i \in \{1,\ldots,n_v\} \tag{12}$$

$$\lim_{p\to 0} \mathbb{E}_{\mathcal{G}_*^{(n_i,k_i)}}[I_{E,C}^{(i)}(p)] = 1 \quad \forall i \in \{1,\ldots,n_c\}. \tag{13}$$

Analogously, (11) is satisfied if

$$\lim_{p\to 1} \mathbb{E}_{\mathcal{G}_*^{(n_i,k_i)}}[I_{E,C}^{(i)}(p)] = 0 \quad \forall i \in \{1,\ldots,n_c\}. \tag{14}$$

To prove (12) it is sufficient to show that for all (n,k) with $k < n$, for any binary matrix in $\mathcal{G}_*^{(n,k)}$ we have $\lim_{p\to 0} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}[I_E(p,q)] = 1 \forall q \in (0,1)$, where $I_E(p,q)$ is given by (2). Moreover, to prove (13) and (14) it is sufficient to show that for all (n,k) with $k < n$, for any binary matrix in $\mathcal{G}_*^{(n,k)}$ we have $\lim_{p\to 0} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}[I_E(p)] = 1$ and $\lim_{p\to 1} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}[I_E(p)] = 0$, respectively, where $I_E(p)$ is given by (3). Consider at first a check component code. From (3) it follows

$$\lim_{p\to 0} I_E(p) = 1 - \left(\tilde{e}_n - \frac{\tilde{e}_{n-1}}{n}\right).$$

Then, the desired property $\lim_{p\to 0} I_E(p) = 1$ is guaranteed by the equality $n\,\tilde{e}_n = \tilde{e}_{n-1}$. This equality is satisfied when the $(k \times n)$ generator matrix $\boldsymbol{G}$ of the check component code is full-rank $(\mathrm{rank}(\boldsymbol{G}) = k)$, and when the $(k \times (n-1))$ matrix obtained by removing any column from $\boldsymbol{G}$ is full rank. In fact, in this case both sides of the previous equality are equal to $k\,n$. By reasoning in the same way, it is readily shown from (3) that

$$\lim_{p\to 1} I_E(p) = 1 - \tilde{e}_1/n.$$

Then, $\lim_{p\to 1} I_E(p) = 0$ when $\tilde{e}_1 = n$, i.e., when the generator matrix of the check component code has no all-zero columns. This is equivalent to assume that the component code has no idle bits, an hypothesis already implicitly considered in (3). We conclude that (13) and (14) (and then (10) and (11)) are satisfied if the generator matrix of the generic check component code is full rank, has no independent columns, and has no zero columns.

Consider a variable component code. From (2):

$$\lim_{p\to 0} I_E(p,q) = 1 - \sum_{z=0}^{k} q^z (1-q)^{k-z} \left(\tilde{e}_{n,k-z} - \frac{\tilde{e}_{n-1,k-z}}{n}\right).$$

If $n\,\tilde{e}_{n,h} = \tilde{e}_{n-1,h}$ for $h = 0,\ldots,k$, then $\lim_{p\to 0} I_E(p,q) = 1 \forall q \in (0,1)$. This is always true when the $(k \times n)$ generator matrix $\boldsymbol{G}$ of the variable component code is full rank $(\mathrm{rank}(\boldsymbol{G}) = k)$ and has no independent columns. In fact, in this case $n\,\tilde{e}_{n,h} = \tilde{e}_{n-1,h} = \binom{k}{h} n\,k$. The constraint that the generator matrix has no zero columns must be also considered, since it is a key hypothesis for the validity of (1). We conclude that (12) (and then (9)) is satisfied if the generator matrix of the generic variable component code is full rank, has no independent columns, and has no zero columns. □

Our aim is to perform a D-GLDPC code threshold analysis through an EXIT chart approach, using the expected EXIT functions expressed by (7) and (8). To apply EXIT chart analysis, we require that the conditions (9), (10) and (11) are satisfied. Conditions (9) and (10) guarantee that, for both the VN set and the CN set, the outcoming average extrinsic information is equal to 1 (i.e., it assumes its maximum value) when the extrinsic channel in Fig. 2 is noiseless. Condition (11) guarantees a zero average extrinsic information outcoming from the CN set when the extrinsic channel is the useless channel. Theorem 2 ensures fulfilling of (9), (10) and (11) when the generator matrix of each VN and CN is drawn from $\mathcal{G}_*^{(n,k)}$ (for proper values of n and k). On the other hand, it is readily shown that the above conditions would not be satisfied for generator matrices randomly drawn from $\mathcal{M}^{(n,k)}$ or $\mathcal{G}^{(n,k)}$ instead of $\mathcal{G}_*^{(n,k)}$.

B. Further Characterization of $\mathcal{G}_*^{(n,k)}$

The expurgated ensemble $\mathcal{G}_*^{(n,k)}$ is obtained by removing from $\mathcal{G}^{(n,k)}$ those matrices either with all-zero columns or with independent columns. Next, we show that removing from $\mathcal{G}^{(n,k)}$ the $(k \times n)$ matrices with independent columns is equivalent to removing from $\mathcal{G}^{(n,k)}$ those matrices representing (n,k) codes with minimum distance 1.

Lemma 1: Let us consider an (n,k) binary linear block code, and let $\boldsymbol{G}$ be any representation of its generator matrix. If $\boldsymbol{G}$ has an independent set of size j, then the code minimum distance satisfies $d_{\min} \le j$.

Proof: Let us suppose that j columns of $\boldsymbol{G}$ form an independent set of size j. For $j \le n - k$, consider the $(k \times (n-j))$ matrix obtained by removing the j columns forming the independent set.[5] Since the rank of this matrix is less than

[5]For $j > n - k$ the lemma is a straightforward consequence of the Singleton bound $d_{\min} \le n - k + 1$.

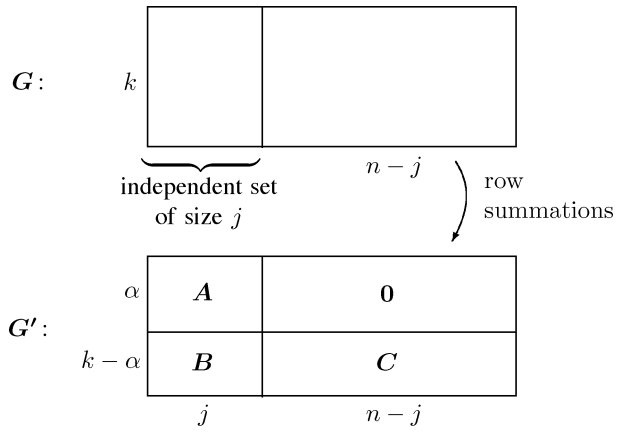

Fig. 3. Two representations of the generator matrix of a linear block code, whose first j columns form an independent set of size j.

$\mathrm{rank}(\boldsymbol{G}) = k$, it is possible to obtain α all-zero rows by row additions only, $\alpha \geq 1$. Applying the same row additions to $\boldsymbol{G}$ provides a new generator matrix representation, that we denote by $\boldsymbol{G}'$, where these α rows[6] have all their 1's lying only in the columns of the independent set (see for example Fig. 3, where the first j columns are assumed to form an independent set, and where $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are nonzero matrices). Any of these α rows is a valid codeword, so that $d_{\min} \leq j$. □

Lemma 2: Let us consider an (n,k) binary linear block code, and let $\boldsymbol{G}$ be any representation of its generator matrix. Then, the following statements are equivalent:

a) the code has minimum distance t;
b) the minimum size of the independent sets of $\boldsymbol{G}$ is t.

Proof: [a $\Rightarrow$ b] If $d_{\min} = t$, then it is possible to construct a representation of $\boldsymbol{G}$ where there is at least one row with exactly t 1's. The columns of $\boldsymbol{G}$ corresponding to these t 1's are an independent set (of size t), because removing them from $\boldsymbol{G}$ leads to a reduction of the rank. This independent set must be of minimum size. In fact, if it existed an independent set of size $j < t$, then from Lemma 1 it would follow $d_{\min} < t$, thus violating the hypothesis $d_{\min} = t$.

[b $\Rightarrow$ a] Let us suppose that the minimum size of the independent sets of $\boldsymbol{G}$ is t, and let us consider an independent set of size t. From Lemma 1 it follows that $d_{\min} \leq t$. The proof is completed by showing that it is not possible to have $d_{\min} < t$. In fact, if $d_{\min} = j < t$ then, by reasoning in the same way as for the [a $\Rightarrow$ b] proof, it would follow that the minimum size of the independent sets of $\boldsymbol{G}$ is $j < t$, which violates the hypothesis. □

Theorem 3: The ensemble $\mathcal{G}_*^{(n,k)}$ is the ensemble of all the $(k \times n)$ binary matrices that represent (n,k) linear block codes without idle bits and with minimum distance at least 2.

[6]Even if not essential for the proof, we observe that $\alpha \leq j$, which can be readily shown as follows. Let $\boldsymbol{M}$ be the matrix obtained by removing the j columns forming the independent set. Then: $\mathrm{rank}(\boldsymbol{M}) \geq k - j$ (as removing j columns can reduce the rank at most by j) and $\mathrm{rank}(\boldsymbol{M}) \leq k - \alpha$ (as $\boldsymbol{M}$ has $k - \alpha$ nonzero rows, not necessarily linearly independent). The inequality $\alpha \leq j$ follows.

Proof: Let us consider an (n,k) binary linear block code $\mathcal{C}$ without idle bits and with $d_{\min} \geq 2$, and let $\boldsymbol{G}$ be any generator matrix of $\mathcal{C}$. First, as $\boldsymbol{G} \in \mathcal{G}^{(n,k)}$, we have $\mathrm{rank}(\boldsymbol{G}) = k$. Second, as the code has no idle bits, $\boldsymbol{G}$ has no all-zero columns. Third, it follows from Lemma 2 that the minimum size of an independent set of $\boldsymbol{G}$ is at least 2, so that $\boldsymbol{G}$ has no independent columns. Therefore $\boldsymbol{G} \in \mathcal{G}_*^{(n,k)}$. Conversely, any matrix $\boldsymbol{G} \in \mathcal{G}_*^{(n,k)}$ represents a linear block code $\mathcal{C}$ of length n and dimension k without idle bits. In fact, we have $\mathcal{G}_*^{(n,k)} \in \mathcal{G}^{(n,k)}$ and $\boldsymbol{G}$ has no all-zero columns. Moreover, since $\boldsymbol{G}$ has no independent columns, we have from Lemma 2 that $\mathcal{C}$ has minimum distance at least 2. □

It follows from (3) that the problem of evaluating the expected EXIT function on $\mathcal{G}_*^{(n,k)}$ for an (n,k) CN over the BEC can be completely solved by evaluating the expected information functions on $\mathcal{G}_*^{(n,k)}$. Similarly, it follows from (2) that the problem of evaluating the expected EXIT function on $\mathcal{G}_*^{(n,k)}$ for an (n,k) VN over the BEC can be completely solved by evaluating the expected split information functions on $\mathcal{G}_*^{(n,k)}$. These two problems are addressed in Sections V and VI, respectively.

V. Expected Information Functions Computation

In this section, we present an approach to compute the expected values of the information functions for any random (n,k) linear block code, where the expectation is over the expurgated ensemble $\mathcal{G}_*^{(n,k)}$ of all the (n,k) generator matrices representing codes with minimum distance at least 2. The method is based on some recursive formulas that allow to compute the exact number of binary matrices with specific properties.

Let $\boldsymbol{G}$ be a random generator matrix from $\mathcal{G}_*^{(n,k)}$, and let $\mathcal{S}_g$ be a submatrix of $\boldsymbol{G}$ obtained selecting g columns. The expectation of $\tilde{e}_g$ can be developed as

$$\begin{aligned}\mathbb{E}_{\mathcal{G}_*^{(n,k)}}[\tilde{e}_g] &= \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\sum_{\mathcal{S}_g} \mathrm{rank}(\mathcal{S}_g)\right] \\ &= \sum_{\mathcal{S}_g} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\mathrm{rank}(\mathcal{S}_g)\right] \\ &= \binom{n}{g} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\mathrm{rank}(\mathcal{S}_g)\right] \qquad (15)\end{aligned}$$

where the last equality is due to the fact that, for random matrices in $\mathcal{G}_*^{(n,k)}$, the expectation of the rank when selecting g columns is independent of the specific selected columns. Without loss of generality, in the following we assume that $\mathcal{S}_g$ in (15) is the submatrix composed of the first g columns of $\boldsymbol{G}$. The expectation of $\mathrm{rank}(\mathcal{S}_g)$ in (15) can be further developed as

$$\begin{aligned}\mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\mathrm{rank}(\mathcal{S}_g)\right] &= \sum_{u=1}^{\min\{k,g\}} u \Pr\{\mathrm{rank}(\mathcal{S}_g) = u\} \\ &= \sum_{u=1}^{\min\{k,g\}} u \frac{K(k,n,g,u,k)}{J(k,n,k)} \qquad (16)\end{aligned}$$

where the summation in u is from $u = 1$ and not from $u = 0$ because $\mathcal{S}_g$ has no zero columns by hypothesis. In (16), we denote

by $J(m,n,r)$ the number of rank-r $(m \times n)$ binary matrices without zero columns, and such that removing any column does not reduce the rank (i.e., with no independent columns). According to Definition 2, we have

$$J(k,n,k) = |\mathcal{G}_*^{(n,k)}|.$$

The function $K(m,n,g,u,r)$ represents the number of rank-r $(m \times n)$ binary matrices without zero columns, without independent columns, and such that the first g columns have rank u. For any $1 \leq g \leq n$, we have

$$\sum_{u=1}^{\min\{k,g\}} K(m,n,g,u,r) = J(m,n,r). \tag{17}$$

Next we develop recursive formulas for computing $J(\cdot)$ and $K(\cdot)$. Even if $J(\cdot)$ can be expressed in terms of $K(\cdot)$ according to (17), an independent recursive formula for $J(\cdot)$ is presented. In this section and in the next section, we often use the following well-known result [35].

Lemma 3: The number of rank-r $(m \times n)$ binary matrices, denoted by $N(m,n,r)$, is given by

$$N(m,n,r) = \prod_{j=0}^{r-1} \frac{(2^m - 2^j)(2^n - 2^j)}{(2^r - 2^j)}.$$

A. Computation of $J(m,n,r)$

The number $J(m,n,r)$ of rank-r $(m \times n)$ binary matrices without zero columns and without independent columns may be computed as the difference between the total number $F(m,n,r)$ of rank-r $(m \times n)$ binary matrices without zero columns and the number of such matrices with at least one independent column.

Lemma 4: Let $F(m,n,r)$ be the number of rank-r $(m \times n)$ binary matrices without zero columns. Then

$$F(m,n,r) = N(m,n,r) - \sum_{z=1}^{n-r} \binom{n}{z} F(m,n-z,r). \tag{18}$$

Proof: See Appendix II. □

For completeness of the recursion (18) it must be imposed $F(m,n,1) = 2^m - 1$, and $F(m,n,r) = 0$ when at least one of the following conditions is true: $m \leq 0, n \leq 0, r \leq 0, r > \min\{m,n\}$.

Theorem 4: The function $J(\cdot)$ can be recursively evaluated according to

$$\begin{aligned} &J(m,n,r) \\ &\quad = F(m,n,r) - \sum_{j=1}^{r} \binom{n}{j} \left[\prod_{i=0}^{j-1} (2^m - 2^i)\right] 2^{j(r-j)} \\ &\qquad \times J(m-j, n-j, r-j). \end{aligned} \tag{19}$$

Proof: See Appendix II. □

For completeness of the recursion (19) it must be imposed $J(m,n,1) = 2^m - 1$, and $J(m,n,r) = 0$ when at least one of the following conditions is true: $m \leq 0, n \leq 0, r \leq 0, r > \min\{m,n\}$.

B. Computation of $K(m,n,g,u,r)$

In order to develop a formula for computing the number $K(m,n,g,u,r)$ of rank-r binary $(m \times n)$ matrices without zero columns, without independent columns, and such that the first g columns have rank equal to u, we use a method analogous to that one used for the function $J(\cdot)$. Let $M(m,n,g,u,r)$ be the number of rank-r binary $(m \times n)$ matrices without zero columns and such that the first g columns have rank equal to u. Then $K(m,n,g,u,r)$ is equal to the difference between $M(m,n,g,u,r)$ and the number of such matrices with at least one independent column.

Lemma 5: Let $T(m,n)$ be the total number of $(m \times n)$ binary matrices without zero columns, i.e.

$$T(m,n) = \sum_{r=1}^{\min\{m,n\}} F(m,n,r)$$

and let by definition

$$T(m,0) = 1.$$

Then:

$$\begin{aligned} &M(m,n,g,u,r) \\ &\quad = F(m,g,u) \sum_{z=0}^{s} \binom{n-g}{z} T(u,z)\, 2^{u(n-g-z)} \\ &\qquad \times F(m-u, n-g-z, r-u). \end{aligned} \tag{20}$$

Proof: See Appendix II. □

For completeness of the recursion (20), we must impose $M(m,n,g,u,r) = 0$ if one of the following conditions is true: $m \leq 0, n \leq 0, u < 0, r \leq 0, g < 0, g > n, u > \min\{m,g\}, r > \min\{m,n\}, r-u > \min\{n-g, m-u\}, u > r, \{g > 0, u = 0\}, \{g = 0, u > 0\}, \{g = n, u \neq r\}, \{m = n, r = m, g \neq u\}$. Special cases are:

$$\begin{aligned} M(m,n,0,0,r) &= F(m,n,r) \\ M(m,n,n,r,r) &= F(m,n,r) \\ M(m,n,g,m,r) &= F(m,g,m)T(m,n-g) \\ M(m,n,g,r,r) &= F(m,g,r)(2^r - 1)^{n-g} \quad \text{if } n > g. \end{aligned}$$

Theorem 5: The function $K(\cdot)$ can be recursively evaluated according to

$$\begin{aligned} &K(m,n,g,u,r) \\ &\quad = M(m,n,g,u,r) \\ &\qquad - \sum_{j=1}^{r-1} \sum_{l=0}^{\min\{u,j\}} \binom{g}{l} \binom{n-g}{j-l} \left[\prod_{i=0}^{j-1} (2^m - 2^i)\right] 2^{j(r-j)} \\ &\qquad \times K(m-j, n-j, g-l, u-l, r-j). \end{aligned} \tag{21}$$

Proof: See Appendix II. □

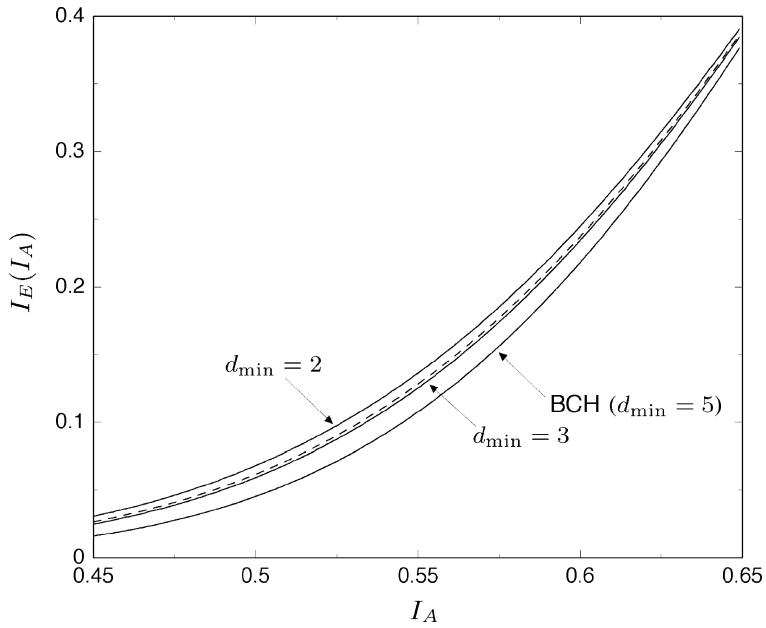

Fig. 4. EXIT functions of a $(31, 21)$ code with $d_{\min} = 2$, a $(31, 21)$ code with $d_{\min} = 3$ and the $(31, 21)$ narrow-sense binary BCH code ($d_{\min} = 5$). Dashed line: Average EXIT function over $\mathcal{G}_*^{(31,21)}$.

For completeness of the recursion (21), we must impose $K(m, n, g, u, r) = 0$ in the same cases where $M(m, n, g, u, r)$ is set to 0. Special cases are

$$\begin{aligned} K(m, n, 0, 0, r) &= J(m, n, r) \\ K(m, n, n, r, r) &= J(m, n, r) \\ K(m, n, g, 1, 1) &= 2^m - 1 \quad \text{if } \{g > 0, n > g\} \\ K(m, m, g, g, m) &= F(m, m, m) \quad \text{if } 1 \le g \le m. \end{aligned}$$

In summary, for some k and n, $E_{\mathcal{G}_*^{(n,k)}}[\tilde{e}_g]$ can be computed from (15) and (16), where $J(\cdot)$ is obtained recursively from (19), and $K(\cdot)$ is obtained recursively from (21).

Example 2: In Fig. 4, a detail of the EXIT function on the BEC for three binary $(31, 21)$ linear block codes (solid lines) is depicted, as a function of the *a priori* mutual information $I_A = 1 - p$, for I_A ranging between 0.45 and 0.65. The minimum distances for the three codes are $2, 3$, and 5, where the $d_{\min} = 5$ code is the $(31, 21)$ narrow-sense binary BCH code. The other two codes were randomly generated. For each of the three codes, the EXIT function has been evaluated by first computing the information functions $\tilde{e}_g$ (which is still feasible for $(31, 21)$ codes, even if time consuming), and then applying (3). In the same figure, the dashed line is the expected EXIT function on $\mathcal{G}_*^{(31,21)}$, evaluated by first computing the expected information functions $\mathbb{E}_{\mathcal{G}_*^{(31,21)}}[\tilde{e}_g]$ according to (15), (16), (19) and (21), and then applying (3). The match between the solid curves and the dashed curve in Fig. 4 is quite good, despite the moderately short codeword length $(n = 31)$. This fact indicates that the expected EXIT function can be confidently used, instead of the exact EXIT function, for longer component codes for which the information functions remain unknown.

VI. Expected Split Information Functions Computation

In this section, the technique for the evaluation of the expected information functions over $\mathcal{G}_*^{(n,k)}$, presented in the previous section, is extended to the split information functions.

Let $\boldsymbol{G}$ be a $(k \times n)$ binary matrix from $\mathcal{G}_*^{(n,k)}$, and let $\mathcal{S}_{g,h}$ be a submatrix of $[\boldsymbol{G} \,|\, \boldsymbol{I}_k]$ obtained by selecting g columns in $\boldsymbol{G}$ and h columns in $\boldsymbol{I}_k$. Then:

$$\begin{aligned} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}[\tilde{e}_{g,h}] &= \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\sum_{\mathcal{S}_{g,h}} \operatorname{rank}(\mathcal{S}_{g,h})\right] \\ &= \sum_{\mathcal{S}_{g,h}} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\operatorname{rank}(\mathcal{S}_{g,h})\right] \\ &= \binom{n}{g}\binom{k}{h} \mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\operatorname{rank}(\mathcal{S}_{g,h})\right]. \quad (22) \end{aligned}$$

The last equality is due to the fact that, for matrices in $\mathcal{G}_*^{(n,k)}$, the expectation of the rank when selecting g columns in $\boldsymbol{G}$ and h columns in $\boldsymbol{I}_k$ is independent of the specific selected columns. Thus, $\mathcal{S}_{g,h}$ in (22) can be in principle any such submatrix.

Without loss of generality, we assume that $\mathcal{S}_{g,h}$ in (22) is the submatrix composed of the last g columns of $\boldsymbol{G}$, and the first h columns of $\boldsymbol{I}_k$ (see Fig. 5). The probability $\Pr\{\operatorname{rank}(\mathcal{S}_{g,h}) = u\}$ that, for a randomly chosen matrix $\boldsymbol{G}$ in $\mathcal{G}_*^{(n,k)}$, the submatrix $\mathcal{S}_{g,h}$ has rank u, can be expressed as the number of matrices in $\mathcal{G}_*^{(n,k)}$ for which this property holds divided by the total number of matrices in $\mathcal{G}_*^{(n,k)}$. It is clear from Fig. 5 that $\operatorname{rank}(\mathcal{S}_{g,h}) \ge h$. In fact, the last h columns of this submatrix, i.e., the first h columns of $\boldsymbol{I}_k$, are linearly independent. Moreover, in order to have $\operatorname{rank}(\mathcal{S}_{g,h}) = u$, it is necessary and sufficient that the $((k - h) \times g)$ submatrix $\boldsymbol{\Gamma}$ in Fig. 5 has rank $u - h$. Hence, the binary matrices $\boldsymbol{G} \in \mathcal{G}_*^{(n,k)}$ leading to $\operatorname{rank}(\mathcal{S}_{g,h}) = u$, are those for which $\operatorname{rank}(\boldsymbol{\Gamma}) = u - h$. Equivalently, they are those

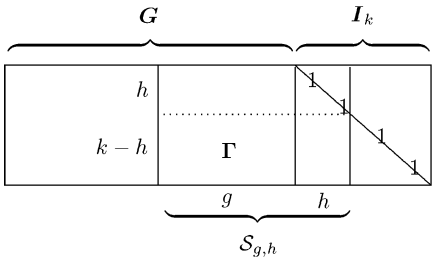

Fig. 5. Definition of the submatrix $\mathcal{S}_{g,h}$ for the evaluation of the expected split information functions.

for which the submatrix intersection of the first g columns and the first $k-h$ rows of $\boldsymbol{G}$ has rank $u-h$ (since $\boldsymbol{\Gamma}$ may in principle be any intersection of $k-h$ rows and g columns of $\boldsymbol{G}$).

Then, the expectation of $\mathrm{rank}(\mathcal{S}_{g,h})$ in (22) can be further developed as

$$\begin{aligned}&\mathbb{E}_{\mathcal{G}_*^{(n,k)}}\left[\mathrm{rank}(\mathcal{S}_{g,h})\right]\\&\quad=\sum_{u=h}^{\min\{k,g+h\}} u\Pr\left\{\mathrm{rank}(\mathcal{S}_{g,h})=u\right\}\\&\quad=\sum_{u=h}^{\min\{k,g+h\}} u\,\frac{\tilde{K}(k,n,g,k-h,u-h,k)}{J(k,n,k)}\end{aligned}\tag{23}$$

where $\tilde{K}(m,n,a,b,t,r)$ represents the number of rank-r $(m\times n)$ binary matrices without zero columns, without independent columns, and such that the submatrix intersection of the first a columns and first b rows has rank t. Note that the function $\tilde{K}(\cdot)$ is a generalization of the function $K(\cdot)$ investigated in the previous section. In fact, we have

$$K(m,n,g,u,r)=\tilde{K}(m,n,g,m,u,r).$$

In the following, a technique for the computation of $\tilde{K}(\cdot)$ is derived.

A. Computation of $\tilde{K}(m,n,a,b,t,r)$

Let $\hat{K}(m,n,a,b,t,u,r)$ be the number of rank-r $(m\times n)$ binary matrices without zero columns, without independent columns, such that the submatrix intersection of the first a columns and b rows has rank t, and such that the submatrix composed of the first a columns has rank u. The function $\tilde{K}(\cdot)$ can be expressed in terms of $\hat{K}(\cdot)$, as

$$\tilde{K}(m,n,a,b,t,r)=\sum_{u=1}^{\min\{m,a\}}\hat{K}(m,n,a,b,t,u,r).\tag{24}$$

The technique for the evaluation of $\tilde{K}(\cdot)$ is based on a recursive formula developed for $\hat{K}(\cdot)$, and presented in the next subsection. For some (m,n,a,b,t,r), the expression for $\hat{K}(m,n,a,b,t,u,r)$ is first evaluated for all $u\in\{1,\ldots,\min\{m,a\}\}$ and then $\tilde{K}(m,n,a,b,t,r)$ is computed according to (24).

B. Computation of $\hat{K}(m,n,a,b,t,u,r)$

In this section a recursion for the computation of $\hat{K}(m,n,a,b,t,u,r)$ is developed.

Lemma 6: Let $\tilde{N}(m,n,p,t,r)$ be the number of rank-r $(m\times n)$ binary matrices, such that the rank of the first p rows is t. Then

$$\begin{aligned}&\tilde{N}(m,n,p,t,r)\\&\quad=2^{t(m-p)}\prod_{j=0}^{t-1}\frac{(2^p-2^j)(2^n-2^j)}{2^t-2^j}\\&\qquad\times\prod_{j=0}^{r-t-1}\frac{(2^{m-p}-2^j)(2^{n-t}-2^j)}{2^{r-t}-2^j}.\end{aligned}\tag{25}$$

Proof: See Appendix III. □

The function $\tilde{N}(\cdot)$ is set to 0 if at least one of the following conditions is true: $m\le 0, n\le 0, p<0, r<0, t<0, t>r, p>m, \{p=0, t>0\}, \{p=m, t\ne r\}, r>\min\{m,n\}, t>\min\{p,n\}$. Particular cases are

$$\tilde{N}(m,n,p,0,r)=\prod_{j=0}^{r-1}\frac{(2^{m-p}-2^j)(2^n-2^j)}{(2^r-2^j)}$$

$$\tilde{N}(m,n,m,r,r)=\prod_{j=0}^{r-1}\frac{(2^m-2^j)(2^n-2^j)}{(2^r-2^j)}.$$

Lemma 7: Let $\tilde{F}(m,n,p,t,r)$ be the number of rank-r $(m\times n)$ binary matrices, such that the rank of the first p rows is t, and without zero columns. Then

$$\begin{aligned}\tilde{F}(m,n,p,t,r)&=\tilde{N}(m,n,p,t,r)\\&\quad-\sum_{z=1}^{n-r}\binom{n}{z}\tilde{F}(m,n-z,p,t,r).\end{aligned}\tag{26}$$

Proof: See Appendix III. □

For completeness of the recursion (26), we must impose $\tilde{F}(m,n,p,t,r)=0$ in the same cases where $\tilde{N}(m,n,p,t,r)$ is set to 0, or when $r=0$. Special cases are:

$$\tilde{F}(m,n,p,0,r)=F(m-p,n,r)$$
$$\tilde{F}(m,n,m,r,r)=F(m,n,r).$$

Lemma 8: Let $\hat{M}(m,n,a,b,t,u,r)$ be the number of rank-r $(m\times n)$ binary matrices without zero columns, such that the submatrix intersection of the first a columns and the first b rows has rank t, and such that the submatrix composed of the first a columns has rank u. Then

$$\begin{aligned}&\hat{M}(m,n,a,b,t,u,r)\\&\quad=\tilde{F}(m,a,b,t,u)\sum_{z=0}^{(n-a)-(r-u)}\binom{n-a}{z}\\&\qquad\times T(u,z)\,2^{u(n-a-z)}\\&\qquad\times F(m-u,n-a-z,r-u)\end{aligned}\tag{27}$$

where $T(m,n)$ is defined as in Lemma 5.

Proof: See Appendix III. □

For completeness of the recursion (27), we must impose $\hat{M}(m,n,a,b,t,u,r)=0$ if at least one of the following

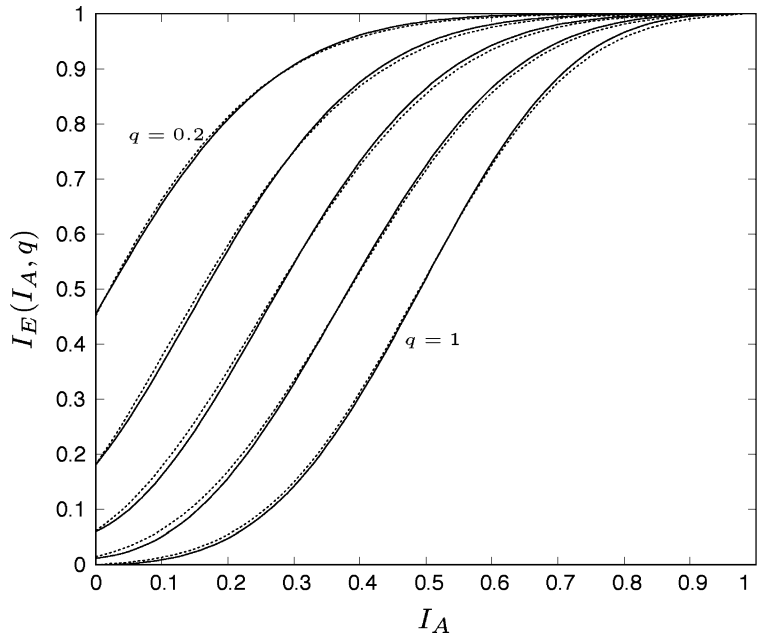

Fig. 6. Comparison between the EXIT function of a $(16, 8)$ variable node, with generator matrix randomly chosen from $\mathcal{G}_*^{(16,8)}$ (solid), and the expected EXIT function over $\mathcal{G}_*^{(16,8)}$ (dotted), for $q \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$.

capacity-approaching distributions. Finally, in Section VII.C capacity-approaching D-GLDPC codes are compared to capacity-approaching LDPC and GLDPC codes, in terms of both asymptotic threshold and finite length performance of long random codes. The obtained results reveal that random D-GLDPC codes can outperform standard LDPC codes and GLDPC codes in terms of asymptotic threshold, waterfall performance and error floor.

A. GLDPC Codes With Uniform Check Node Structure

Let us consider a GLDPC code with $(31, 21)$ BCH codes as CNs and length-2 repetition codes as VNs. The code rate is $R = 0.35484$, corresponding to a Shannon limit over the BEC equal to $1 - R = 0.64516$. Let us assume d-bounded distance decoding (see Section III-C) at the BCH CNs. The GLDPC code threshold can be evaluated with the EXIT chart based on (4), by numerically evaluating the $(31, 21)$ BCH code information functions. The EXIT functions for $d \in \{4, 7, 10, 31\}$ are depicted in Fig. 7 (solid curves) as a function of the extrinsic channel erasure probability p. The corresponding GLDPC thresholds are given in Table I. Next, let us consider the same class of GLDPC codes, under the hypothesis that the $(31, 21)$ CNs are random linear block codes from $\mathcal{G}_*^{(31,21)}$. The corresponding expected EXIT functions are depicted in Fig. 7 (dotted curves), and the GLPDC thresholds under p-bounded-distance decoding are given in Table I for $d \in \{4, 7, 10, 31\}$.

The threshold values in Table I suggest the following. From a threshold point of view, when the maximum number d of erasures faced by the decoder is small, it is convenient to use a check component code with a good minimum distance, like the BCH code. On the contrary, for a higher d, or if no bound on d is imposed at all ($d = 31$, which corresponds to MAP decoding), linear block codes must exist within $\mathcal{G}_*^{(n,k)}$ that guarantee a better GLDPC threshold than the BCH code. In fact, for sufficiently high d, the threshold computed assuming the expected CN set EXIT function is better than the threshold obtained with the BCH CNs. For this specific example, the crossover point between the ensemble average and the BCH code is at $d = 12$.

We actually found $(31, 21)$ linear block codes for which the GLDPC threshold is better than the ensemble average, under unconstrained MAP decoding. For instance, we generated a $(31, 21)$ code with $d_{\min} = 2$ for which the GLDPC threshold is $q^* = 0.51920$. We also generated a $(31, 21)$ linear code characterized by $d_{\min} = 3$, and we found a threshold $q^* = 0.51310$ for the GLDPC code. This value is intermediate w.r.t. the thresholds corresponding to the BCH code and to the $d_{\min} = 2$ code. The EXIT functions for the $d_{\min} = 2$ and $d_{\min} = 3$ codes are those already shown in Fig. 4. Denoting by $T(\mathcal{C})$ the GLDPC code threshold corresponding to the choice of component code $\mathcal{C}$ for the CNs, we have $T(\mathcal{C}_{\text{BCH}}^{(31,21)}) < T(\mathcal{C}_{d_{\min}=3}^{(31,21)}) < T(\mathbb{E}[\mathcal{C}^{(31,21)}]) < T(\mathcal{C}_{d_{\min}=2}^{(31,21)})$. This reveals that using weak codes as check component codes for GLDPC codes with a uniform check structure can be more favorable, from a threshold viewpoint, than using more powerful codes, like the BCH codes. This fact is confirmed by the simulation result shown in Fig. 8, in which the waterfall performance of a $(3999, 1419)$ GLDPC code using $(31, 21)$ BCH CNs is worse than the waterfall performance of a GLDPC code, having the same bipartite graph, and using the $d_{\min} = 2$ code as check component code.

B. Capacity-Approaching GLDPC Codes With Hybrid Check Node Structure

Let us first consider the problem of finding the LDPC degree distribution with the largest threshold over the BEC and subject to the following constraints: VN degrees ranging from 2 up to 30, CN degrees ranging from 3 up to 14, code rate $R = 1/2$.

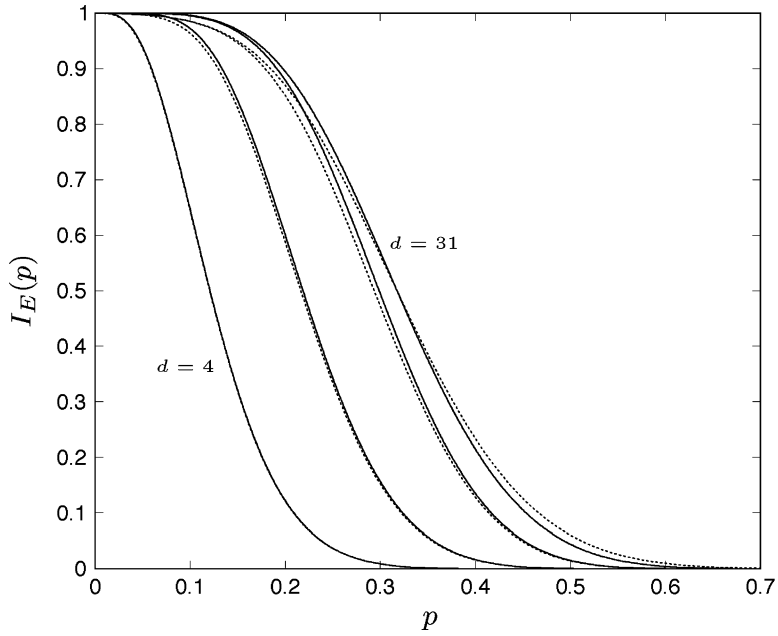

Fig. 7. EXIT function of the $(31, 21)$ BCH code (solid) and expected EXIT function over the expurgated ensemble $\mathcal{G}_*^{(31,21)}$ (dotted) under d-bounded-distance decoding, for $d \in \{4, 7, 10, 31\}$.

TABLE I
THRESHOLDS OVER THE BEC FOR GLPDC CODES WITH $(31, 21)$ BCH CNS AND THRESHOLDS EVALUATED WITH THE EXPECTED EXIT FUNCTION OVER $\mathcal{G}_*^{(31,21)}$

d	BCH	expectation
4	0.21915	0.21879
7	0.35596	0.35407
10	0.46256	0.45929
31	0.50187	0.51426

We solved the problem using the differential evolution (DE) algorithm [36]. DE is an evolutionary, parallel optimization algorithm to find the global minimum of a real-valued function of a vector of continuous parameters. The algorithm is based on the evolution of a population of N_p vectors, and its main steps are similar to those of evolutionary optimization algorithms [37]. Once a starting population of N_p vectors has been generated (*initialization*), a competitor for each population element is generated by properly combining a subset of randomly chosen vectors from the same population (*mutation* and *crossover*). Each element of the population is then compared with its competitor : the vector yielding a smaller value of the cost function is selected (*selection*) as an element of the evolved population. The mutation, crossover and selection steps are iterated until a certain stopping criterion is fulfilled.[8] Introduced in [38], DE was first proposed for the optimization of LDPC codes degree profile in [39]. In this specific case, each element of the population is a degree distribution pair, while the cost function is the function returning the threshold of a degree distribution pair.[9]

Differential evolution was run with $N_p = 70$, for different initial populations. The threshold of the best found distribution is $q^* = 0.49611$, quite close to the Shannon limit $1 - R = 0.5$. The distribution is described in Table II (LDPC column).

Next, we solved the same optimization problem for a GLDPC code with a hybrid CN structure, composed of SPC codes and $(31, 21)$ linear block codes. We solved again the optimization problem with the DE algorithm, assuming the same degree constraints for the VNs and for the SPC CNs, and again $R = 1/2$. More specifically, we separately solved the problem in the cases where the $(31, 21)$ CNs are represented by the binary BCH code with $d_{\min} = 5$ and by the $d_{\min} = i$ codes $(i = 2, 3)$ considered in previous subsection. We also solved the optimization problem using the expected EXIT function over the expurgated ensemble $\mathcal{G}_*^{(31,21)}$. The optimal distribution corresponding to the choice of the $(31, 21)$ BCH code is described in Table II (GLDPC column), while the four GLDPC thresholds are compared in Table III.

For all choices of the $(31, 21)$ generalized CNs, the edges for the capacity-approaching distribution are connected to the SPC nodes with degree 9 and to the $(31, 21)$ nodes. Moreover, the optimized distributions (both variable and check) are very similar in all cases. The fraction of edges connected to the $(31, 21)$ codes ranges from about 5.73% for the $d_{\min} = 2$ code to about 8.72% for the BCH code. Some considerations about these results are presented next.

First, in this case study it has been possible to improve the threshold of the LDPC code by letting unchanged the constraints on the degrees of the repetition and SPC nodes, introducing check component codes different from the SPC codes, and prop-

[8] In evolutionary algorithms, the weakest elements of the population are typically replaced by stronger mutant elements. On the other hand, in DE a competitor is created for each vector of the population, and compared only with that vector. Heuristically, this choice is effective to avoid that the algorithm remains trapped in local minima. It is also worth mentioning the peculiar mutation technique of DE, where a mutant vector is obtained by adding to a vector the difference (multiplied by a scaling factor) between two other vectors.

[9] A threshold sign change is necessary, for instance, over the BEC as we look for the global maximum.

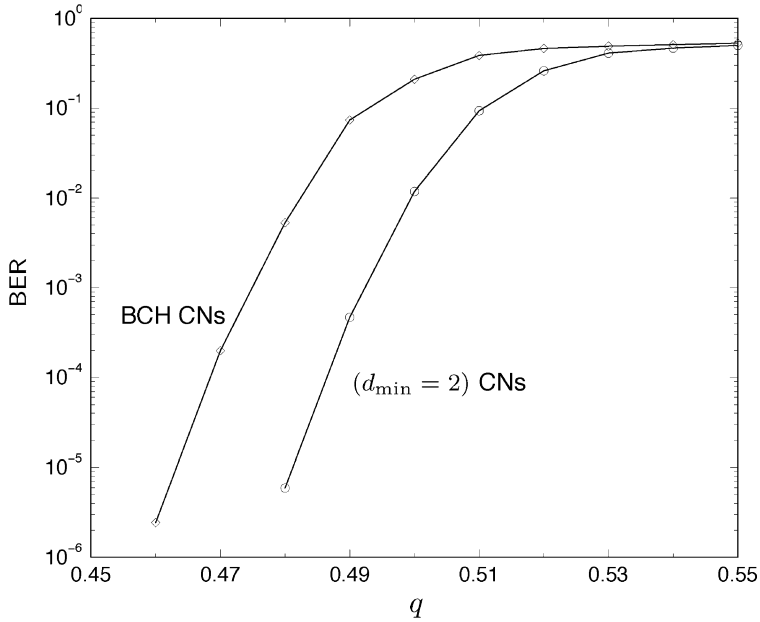

Fig. 8. Comparison between the waterfall performance of a $(3999, 1419)$ GLDPC code with uniform CN structure composed of $(31, 21)$ binary BCH codes and a $(3999, 1419)$ GLDPC code with uniform CN structure composed of $(31, 21)$ linear block codes with $d_{\min} = 2$. The bipartite graph is the same for the two GLDPC codes.

TABLE II
CAPACITY-APPROACHING RATE-$1/2$ LDPC, GLPDC, D-GLDPC$_1$ AND D-GLDPC$_2$ EDGE DISTRIBUTIONS

Code type	LDPC	GLDPC	D-GLDPC$_1$	D-GLDPC$_2$
Variable Nodes				
Rep. 2	0.281884	0.270712	0.287410	0.280377
Rep. 3	0.123242	0.168858	0.161606	0.156363
Rep. 4	0.060701			
Rep. 5	0.106412	0.165958	0.293870	0.358294
Rep. 8		0.230227		
Rep. 9	0.084976			
Rep. 10	0.103547			
Rep. 29			0.217547	
Rep. 30	0.239238	0.164246		0.174163
$(31, 10)$			0.039568	0.030803
Check Nodes				
SPC 8	0.925027			
SPC 9		0.912838	0.871398	0.381799
SPC 10	0.074973			0.458201
$(31, 21)$-BCH		0.087162	0.128602	0.160000
Thresholds				
q^*	0.49611	0.49671	0.49759	0.49655

TABLE III
THRESHOLDS q^* FOR CAPACITY-APPROACHING RATE-1/2 LDPC AND GLPDC DISTRIBUTIONS

LDPC	0.49611
GLDPC, $d_{\min} = 2$	0.49627
GLDPC, expectation	0.49639
GLDPC, $d_{\min} = 3$	0.49648
GLDPC, BCH	0.49671

erly modifying the edge distribution. The presented example is even more meaningful, since the starting LDPC distribution is already capacity-approaching. This better GLDPC threshold has been achieved with a relatively small fraction of generalized CNs: the fraction of BCH CNs is about 2.70%, which results in a small increase in terms of decoding complexity w.r.t. the LDPC code.

Second, when considering hybrid CN structures instead of uniform ones, using more powerful codes like the BCH codes (judiciously mixed with SPC codes) leads to better thresholds. For the hybrid case we obtain $T(\mathcal{C}_{d_{\min}=2}^{(31,21)}) < T(\mathbb{E}[\mathcal{C}^{(31,21)}]) < T(\mathcal{C}_{d_{\min}=3}^{(31,21)}) < T(\mathcal{C}_{\mathrm{BCH}}^{(31,21)})$, which is the opposite of what was found for a uniform check node structure. The reason is that the role of weak codes (necessary for obtaining good thresholds) is now played by the SPC codes.

Third, when combined with DE, the developed technique for the expected EXIT function of generalized CNs in $\mathcal{G}_*^{(n,k)}$ leads to an optimal distribution and threshold which closely match those obtained for the choice of the BCH CNs. Hence, this technique can be confidently used not only for the threshold analysis, but also for the purposes of GLDPC distribution optimizations.

C. Capacity Approaching D-GLDPC Codes

We solved the same optimization problem as that considered in the previous subsection for a $R = 1/2$ D-GLDPC coding scheme. Generalized $(31, 21)$ BCH CNs and SPC CNs were considered; in addition, a hybrid VN set was allowed, composed by a mixture of repetition codes with the same degrees as for the LDPC code, with the addition of $(31, 10)$ random linear block codes from $\mathcal{G}_*^{(31,10)}$. The choice of $(31, 10)$ codes as VNs was dictated by the heuristic guideline to use codes with same length and dual dimensions at opposite sides of the bipartite graph. The random code approach was followed since the direct computation of the split information functions for a specific $(31, 10)$ linear block code, e.g., the dual of the $(31, 21)$ BCH code, is not feasible in terms of computation time. The expected EXIT

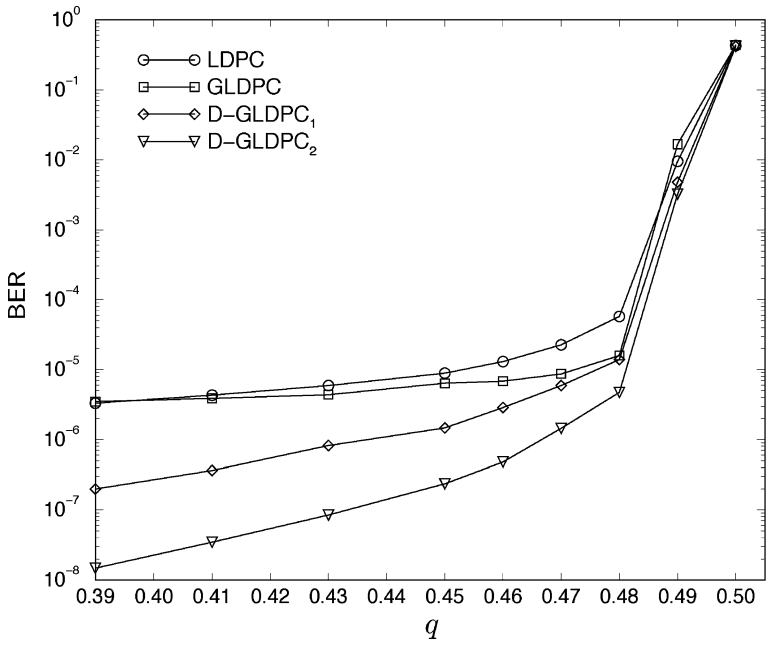

Fig. 9. Performance of $R = 1/2$ LDPC, GLDPC and D-GLDPC codes of length $N = 128000$ on the BEC.

function for the generalized VNs over $\mathcal{G}_*^{(31,10)}$ was evaluated according to the method presented in Section V. The capacity-approaching D-GLDPC distribution obtained by DE is shown in Table II, together with its threshold, and called D-GLDPC$_1$ distribution. Some considerations are provided next.

First, the D-GLDPC$_1$ distribution has the best threshold. Hence, under the described constraints, using generalized VNs together with generalized CNs increases the threshold w.r.t. GLDPC codes, even in a case study where the GLDPC threshold is already very close to capacity. This better D-GLDPC threshold is achieved with a small increase of the fraction of BCH CNs, and with a small fraction of generalized VNs. In fact, the fraction of BCH CNs and $(31, 10)$ VNs for the D-GLDPC$_1$ code are about 4.11% and 0.48%, respectively, which results in a small increase in decoding complexity w.r.t. the GLDPC code.

Second, the larger fraction of BCH CNs in the D-GLDPC$_1$ distribution than in the GLDPC one (4.11% versus 2.70%) suggests the following. The original idea behind GLDPC codes was to strengthen the CN set, by introducing powerful generalized CNs [12]. This approach can provide good minimum distance properties, but the drawback is a lowering of the overall code rate, which reveals unacceptable in many cases [18]. In the case study under analysis, the introduction of $(31, 10)$ generalized VNs is able to partly compensate the rate loss due to the $(31, 21)$ BCH CNs. Then it is possible to use a larger number of powerful erasure correcting codes at the CNs, with no threshold loss.

In order to support these asymptotic results, we simulated long and randomly constructed codes, designed according to the distributions presented in Table II. Random codes were simulated, because random connections between the VN set and the CN set are assumed in (5) and (6). For the D-GLDPC coding scheme, the dual of the $(31, 21)$ BCH code was used at the generalized VNs. In Fig. 9, the performance in terms of post-decoding bit erasure rate (BER) is shown for codes of length $N = 128000$. As expected, the LDPC code exhibits bad error floor performance, due to the poor minimum distance of capacity approaching distributions [10] (for this distribution we have $\lambda'(0)\rho'(1) = 2.015$). This high error floor is not improved when considering the GLDPC code construction. However, the D-GLDPC code exhibits both a slightly better waterfall performance (according to the slightly better threshold) and an error floor which is about one order of magnitude lower than that of LDPC and GLDPC codes. This result suggests that capacity approaching D-GLDPC codes can be constructed, characterized by better properties in terms of both waterfall and error floor performance than that of LDPC and GLDPC codes, and with limited increase of decoding complexity.

Using generalized VNs enables to use a larger number of generalized (powerful) CNs than for GLDPC codes, providing better minimum distance properties, while keeping a better threshold. In order to construct D-GLDPC codes with better minimum distance properties than the D-GLDPC$_1$ code, and still a good threshold, we tried the following approach. We ran the DE algorithm again for the D-GLDPC distribution, with the additional constraint of a lower bound on the fraction of edges incident on the generalized CNs. More specifically, we ran the DE optimization with the further constraint $\rho_{\mathrm{BCH}} \geq \rho_{\mathrm{BCH}}^{\min}$. The obtained distribution for $\rho_{\mathrm{BCH}}^{\min} = 0.16$ and the corresponding threshold are presented in Table II, in the D-GLDPC$_2$ column. The threshold is still better than that of the LDPC distribution.

The performance curve on the BEC obtained for a random $N = 128000$ code, designed according to the D-GLDPC$_2$ distribution, is also shown in Fig. 9. We observe an improvement in the error floor region, about one order of magnitude w.r.t. the D-GLDPC$_1$ code, and about two orders of magnitude w.r.t. the GLDPC and LDPC codes, with no loss in terms of waterfall performance.

VIII. Conclusion

In this paper, a technique for the asymptotic analysis of D-GLDPC codes on the BEC has been proposed. This technique assumes that the variable and check component codes are random codes. It computes the expected EXIT function for the variable and check node decoders, thus enabling an EXIT chart analysis. The core of this method is the computation of the expected (split) information functions over an expurgated ensemble of (n,k) linear block codes. The expurgation guarantees a correct application of the EXIT charts analysis. The expected (split) information function computation exploits some formulas for obtaining the exact number of binary matrices with specific properties. The proposed analysis method has been combined with the DE algorithm, in order to search good D-GLDPC distributions. Focusing on random capacity approaching codes, it has been shown that D-GLDPC codes can be constructed, outperforming LDPC and GLDPC codes in terms of *both* waterfall and error floor. Moreover, by lower bounding the fraction of edges toward the generalized CNs, D-GLDPC codes have been designed with significantly better error floor than LDPC and GLDPC codes and no sacrifice in terms of waterfall performance.

Appendix I
Proof of Theorem 1

Let us denote by $\boldsymbol{G}$ the generic $(k \times n)$ generator matrix of the CN. If the extrinsic channel is a BEC we have $E_i \in \{-\infty, +\infty, 0\}$, where $E_i = 0$ corresponds to an erasure message. Therefore

$$\begin{aligned} I_E(p) &\triangleq \frac{1}{n}\sum_{i=1}^{n} I(V_i;E_i) \\ &= \frac{1}{n}\sum_{i=1}^{n} H(V_i) - \frac{1}{n}\sum_{i=1}^{n} H(V_i|E_i) \\ &\overset{(a)}{=} 1 - \frac{1}{n}\sum_{i=1}^{n} H(V_i|E_i=0)\Pr\{E_i=0\} \\ &\overset{(b)}{=} 1 - \frac{1}{n}\sum_{i=1}^{n} \Pr\{E_i=0\} \end{aligned} \tag{30}$$

where (a) follows from $H(V_i \,|\, E_i = -\infty) = H(V_i \,|\, E_i = +\infty) = 0$ and from the hypothesis that the code has no idle bits, and (b) from $H(V_i|E_i = 0) = 1$. Under d-bounded-distance decoding, we have $E_i = 0$ when either the number of erasures in $\boldsymbol{W}_{[i]}$ is larger than or equal to d, or when it is smaller than d but the nonerasure elements of $\boldsymbol{W}_{[i]}$ are not sufficient to recover V_i. Denoting these two disjoint events by $\mathcal{E}_{1,i}$ and $\mathcal{E}_{2,i}$, respectively, (30) may be written as

$$I_E(p) = 1 - \frac{1}{n}\sum_{i=1}^{n}\Pr\{\mathcal{E}_{i,1}\} - \frac{1}{n}\sum_{i=1}^{n}\Pr\{\mathcal{E}_{i,2}\}. \tag{31}$$

It is readily shown that

$$\frac{1}{n}\sum_{i=1}^{n}\Pr\{\mathcal{E}_{i,1}\} = \sum_{t=d}^{n-1}\binom{n-1}{t} p^t(1-p)^{n-1-t}. \tag{32}$$

To develop the third summand in the RHS of (31), let us introduce the random variable T_i as the number of erasures in $\boldsymbol{W}_{[i]}$, and the set $\mathcal{W}_{[i]}$ of all realizations $\boldsymbol{w}_{[i]}$ of $\boldsymbol{W}_{[i]}$ such that the ith column of $\boldsymbol{G}$ (associated with V_i) is linearly independent of the columns of $\boldsymbol{G}$ associated with the nonerasure elements of $\boldsymbol{w}_{[i]}$. We have

$$\begin{aligned} \Pr\{\mathcal{E}_{2,i}\} &= \Pr\{T_i \le d-1, \boldsymbol{W}_{[i]} \in \mathcal{W}_{[i]}\} \\ &= \sum_{t=0}^{d-1} \Pr\{T_i \le d-1, \boldsymbol{W}_{[i]} \in \mathcal{W}_{[i]} | T_i = t\} \\ &\quad \times \Pr\{T_i = t\} \\ &= \sum_{t=0}^{d-1} \Pr\{\boldsymbol{W}_{[i]} \in \mathcal{W}_{[i]} | T_i = t\} \\ &\quad \times \binom{n-1}{t} p^t(1-p)^{n-1-t} \\ &= \sum_{t=0}^{d-1} \mathcal{N}(\hat{\boldsymbol{w}}_{[i]}^{(t)}) p^t (1-p)^{n-1-t} \end{aligned} \tag{33}$$

where the last equality follows from the fact that $\Pr\{\boldsymbol{W}_{[i]} \in \mathcal{W}_{[i]} \,|\, T_i = t\} = \mathcal{N}(\hat{\boldsymbol{w}}_{[i]}^{(t)})/\binom{n-1}{t}$, where $\mathcal{N}(\hat{\boldsymbol{w}}_{[i]}^{(t)})$ is the number of realizations $\boldsymbol{w}_{[i]}$ characterized by t erasures and belonging to $\mathcal{W}_{[i]}$ (denoted by $\hat{\boldsymbol{w}}_{[i]}^{(t)}$). Let $\boldsymbol{G}_{[i]}$ be the matrix $\boldsymbol{G}$ except its ith column. For a given realization $\boldsymbol{w}_{[i]}^{(t)}$ with t erasures, let $\mathcal{S}_{[i],n-1-t}$ be the $(k \times (n-1-t))$ matrix formed by the $n-1-t$ columns of $\boldsymbol{G}_{[i]}$ corresponding to the nonerasure elements of $\boldsymbol{w}_{[i]}^{(t)}$. Moreover, let us define the $(k \times (n-t))$ matrix $\mathcal{S}'_{[i],n-t} = [\mathcal{S}_{[i],n-1-t} | \boldsymbol{g}_i]$, where $\boldsymbol{g}_i$ is the ith column of $\boldsymbol{G}$. Using (33) and denoting by $\sum_{\mathcal{S}_{[i],n-1-t}}$ the summation over all possible matrices $\mathcal{S}_{[i],n-1-t}$, we can write

$$\begin{aligned} &\frac{1}{n}\sum_{i=1}^{n}\Pr\{\mathcal{E}_{i,2}\} \\ &\quad = \frac{1}{n}\sum_{i=1}^{n}\sum_{t=0}^{d-1} \mathcal{N}\left(\hat{\mathbf{w}}_{[i]}^{(t)}\right) p^t(1-p)^{n-1-t} \\ &\quad = \frac{1}{n}\sum_{t=0}^{d-1} p^t(1-p)^{n-1-t} \sum_{i=1}^{n} \mathcal{N}\left(\hat{\mathbf{w}}_{[i]}^{(t)}\right) \\ &\quad \overset{(a)}{=} \frac{1}{n}\sum_{t=0}^{d-1} p^t(1-p)^{n-1-t} \\ &\qquad \times \sum_{i=1}^{n} \sum_{\mathcal{S}_{[i],n-1-t}} \left[\operatorname{rank}\left(\mathcal{S}'_{[i],n-t}\right) - \operatorname{rank}\left(\mathcal{S}_{[i],n-1-t}\right)\right] \\ &\quad \overset{(b)}{=} \frac{1}{n}\sum_{t=0}^{d-1} p^t(1-p)^{n-1-t}[(n-t)\tilde{e}_{n-t} - (t+1)\tilde{e}_{n-1-t}]. \end{aligned} \tag{34}$$

In the previous equation list, (a) follows from the fact that $\operatorname{rank}(\mathcal{S}'_{[i],n-t}) - \operatorname{rank}(\mathcal{S}_{[i],n-1-t})$ equals 1 in correspondence with realizations $\hat{\boldsymbol{w}}_{[i]}^{(t)}$ and equals 0 in correspondence with any other realization $\boldsymbol{w}_{[i]}^{(t)}$; (b) follows from the definition of information function. Substituting (32) and (34) into (31) completes the proof of the theorem.

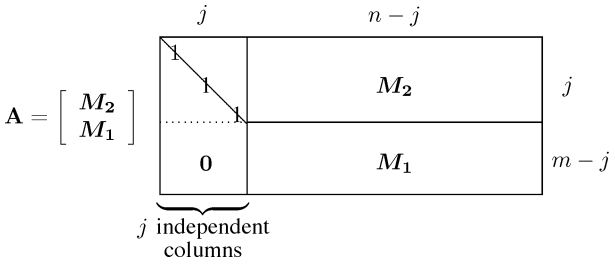

Fig. 10. Specific matrix for the computation of the function $J(\cdot)$.

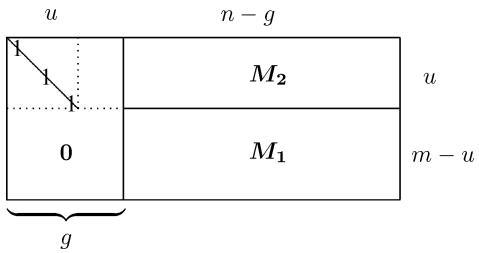

Fig. 11. Specific matrix for the computation of the function $M(\cdot)$.

APPENDIX II
PROOFS OF LEMMAS AND THEOREMS OF SECTION V

Proof of Lemma 4: $F(m,n,r)$ is equal to the total number of rank-r $(m \times n)$ binary matrices, $N(m,n,r)$, minus the number of rank-r $(m \times n)$ binary matrices with at least one zero column. The number of rank-r $(m \times n)$ binary matrices with exactly z zero columns $(z \leq n-r)$ is expressed by $\binom{n}{z} F(m, n-z, r)$. □

Proof of Theorem 4: $J(m,n,r)$ can be computed as $F(m,n,r) - \sum_{j=1}^{r} J^{(j)}(m,n,r)$, where $J^{(j)}(m,n,r)$ is the number of rank-r $(m \times n)$ binary matrices without zero columns and with j independent columns.[10]

There are $\binom{n}{j}$ possible positions for the j independent columns, and the number of choices of the j independent columns is $\prod_{i=0}^{j-1}(2^m - 2^i)$. Hence we have

$$J^{(j)}(m,n,r) = \binom{n}{j} \left[\prod_{i=0}^{j-1}(2^m - 2^i)\right] D^{(j)}(m, n-j)$$

where $D^{(j)}(m, n-j)$ is the residual number of $(m \times (n-j))$ binary matrices (that must have no zero columns and no independent columns). We prove next that

$$D^{(j)}(m, n-j) = J(m-j, n-j, r-j)\, 2^{j(r-j)} \qquad (35)$$

thus leading to the recursion (19).

Since $D^{(j)}(m, n-j)$ is independent of the position and choice of the j independent columns, we can reason on the specific matrix shown in Fig. 10, where the matrix $\boldsymbol{A}$ is defined.

[10]The summation in j can be actually always stopped at $j = r-1$, i.e., $J(m,n,r) = F(m,n,r) - \sum_{j=1}^{r-1} J^{(j)}(m,n,r)$, except for full-rank matrices for which $m \geq n$. Since for binary $(k \times n)$ generator matrices $k < n$ is always assumed, for the purpose of expected information function computation, the summation in j up to $r-1$ is sufficient. This fact is implicitly used in the proofs of Theorem 5 and Theorem 6 as well.

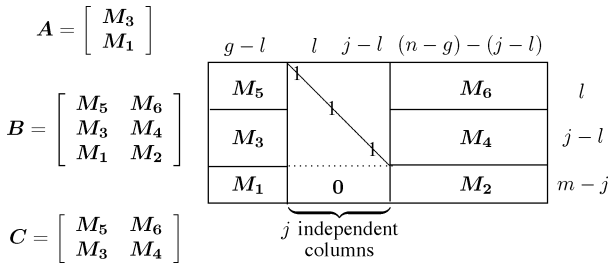

Fig. 12. Matrix for the computation of the function $K(\cdot)$.

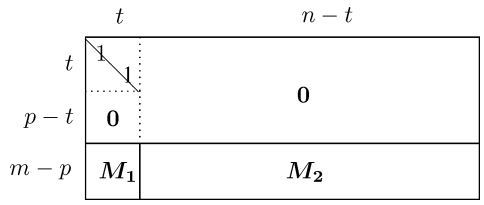

Fig. 13. Specific matrix for the computation of the function $\bar{N}(\cdot)$.

With respect to this choice, $D^{(j)}(m, n-j)$ is the number of choices of the last $n-j$ columns.

The rank of the matrix in Fig. 10 is equal to $j + \mathrm{rank}(\boldsymbol{M_1})$. Thus $\boldsymbol{M_1}$ must have rank $r-j$, and it must have no independent columns. In fact, since the total rank is $j + \mathrm{rank}(\boldsymbol{M_1})$, an independent column for $\boldsymbol{M_1}$ would be independent for the whole matrix. Moreover, since each of the last $n-j$ columns must be linearly independent of each of the first j columns, each column of $\boldsymbol{M_1}$ must have at least one 1. Hence, the number of $\boldsymbol{M_1}$ matrices is equal to $J(m-j, n-j, r-j)$. Since the first j columns are independent columns, removing them from the matrix must lead to a rank $r-j$, so that $\mathrm{rank}(\boldsymbol{A}) = \mathrm{rank}(\boldsymbol{M_1}) = r-j$. Then, any row in $\boldsymbol{M_2}$ must be a linear combination of rows in $\boldsymbol{M_1}$. The total number of such combinations is $2^{j(r-j)}$. Equation (35) follows. □

Proof of Lemma 5: The function $M(m,n,g,u,r)$ can be expressed as $F(m,g,u)$ (number of choices of the first g columns) times the number of $(m \times (n-g))$ binary matrices without zero columns and such that the overall rank is r. Since this number is independent of the specific choice of the first g columns, we can reason on the specific matrix depicted in Fig. 11.

In order to have an overall rank r, we must have $\mathrm{rank}(\boldsymbol{M_1}) = r-u$. Denoting by z the number of zero columns in $\boldsymbol{M_1}$, the number of $\boldsymbol{M_1}$ matrices can be expressed as $\binom{n-g}{z} F(m-u, n-g-z, r-u)$. Since $\mathrm{rank}(\boldsymbol{M_1}) = r-u$, the number z of zero columns in $\boldsymbol{M_1}$ cannot exceed $s = (n-g) - (r-u)$. The only constraint on $\boldsymbol{M_2}$ is that at least one 1 must be present in each column of $\boldsymbol{M_2}$ corresponding to a zero column of $\boldsymbol{M_1}$. Thus the number of $\boldsymbol{M_2}$ matrices corresponding to a $\boldsymbol{M_1}$ matrix with z zero columns is $T(u,z)\, 2^{u(n-g-z)}$, where $T(\cdot)$ is defined in the statement of the lemma, and where $T(u,0)$ must be set to 1 (no zero columns in $\boldsymbol{M_1}$). Then we obtain (20). □

Proof of Theorem 5: In a similar way as for the function $J(\cdot)$, we have $K(m,n,g,u,r) = M(m,n,g,u,r) - \sum_{j=1}^{r-1} K^{(j)}(m,n,g,u,r)$, where $K^{(j)}(m,n,g,u,r)$ is the number of rank-r $(m \times n)$ binary matrices without zero

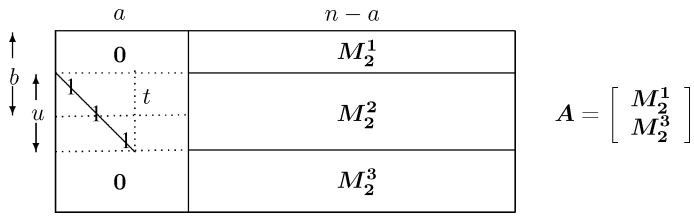

Fig. 14. Specific choice of the submatrix $\boldsymbol{M_1}$ for the computation of $\hat{M}(m,n,a,b,t,u,r)$.

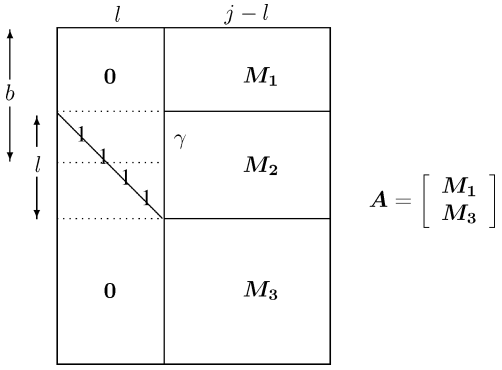

Fig. 15. Specific choice of the $(m \times j)$ matrix for the computation of $G(m,j,l,b,\gamma,\delta)$.

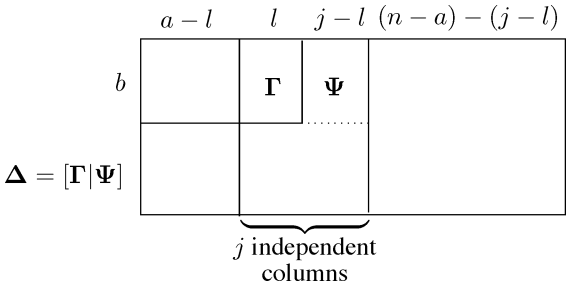

Fig. 16. Specific position of the j independent columns, with $\mathrm{rank}(\boldsymbol{\Gamma}) = \gamma$ and $\mathrm{rank}(\boldsymbol{\Delta}) = \delta$.

columns, with the first g columns having a rank u, and with j independent columns.

Let the number of independent columns among the first g columns be $l \leq \min\{u,j\}$, and the number of independent columns among the last $n-g$ columns be $j-l$. The number of possible positions of the independent columns is $\binom{g}{l}\binom{n-g}{j-l}$, while the number of choices of the j independent columns is $\prod_{i=0}^{j-1}(2^m-2^j)$. We can reason on a specific position and choice of the independent columns. This specific choice is depicted in Fig. 12, where the matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are defined. We have

$$K^{(j)}(m,n,g,u,r) = \sum_{l=0}^{\min\{u,j\}} \binom{g}{l}\binom{n-g}{j-l}\left[\prod_{i=0}^{j-1}(2^m-2^i)\right] N_B$$

with N_B defined as the number of $\boldsymbol{B}$ matrices for each choice of the j independent columns. We prove next that the number of $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$ possible matrices is $K(m-j,n-j,g-l,u-l,r-j)$ and, for each choice of $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$, the number of $\boldsymbol{C}$ matrices is $2^{j(r-j)}$, so that

$$N_B = 2^{j(r-j)}\, K(m-j,n-j,g-l,u-l,r-j)$$

from which the recursion (21) follows.

The rank of the overall matrix is equal to $j+\mathrm{rank}([\boldsymbol{M_1}|\boldsymbol{M_2}])$. Consequently, $\mathrm{rank}([\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]) = r-j$. Furthermore, $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$ must have no independent columns. In fact, since the overall rank is $j + \mathrm{rank}([\boldsymbol{M_1} \,|\, \boldsymbol{M_2}])$, any such column would also be an independent column for the overall matrix. The matrix $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$ must also have at least one 1 for each column due to the linear independence between the j independent columns and all the columns of $\boldsymbol{B}$. Finally, $\mathrm{rank}(\boldsymbol{M_1}) = u-l$. This latter condition can be obtained in the following way. Since $\mathrm{rank}(\boldsymbol{B}) = \mathrm{rank}([\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]) = r-j$, each row in $\boldsymbol{C}$ must be a linear combination of rows in $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$. This implies in particular that each row in $\boldsymbol{M_3}$ must be a linear combination of rows in $\boldsymbol{M_1}$, i.e., $\mathrm{rank}(\boldsymbol{A}) = \mathrm{rank}(\boldsymbol{M_1})$. The rank of the first g columns is equal to $l + \mathrm{rank}(\boldsymbol{A})$. Since this rank must be equal to u, it follows $\mathrm{rank}(\boldsymbol{A}) = u-l$, i.e., $\mathrm{rank}(\boldsymbol{M_1}) = u-l$. Then, the number of $[\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]$ matrices is $K(m-j,n-j,g-l,u-l,r-j)$.

Since each row in $\boldsymbol{C}$ is a linear combination of rows of $[\boldsymbol{M_1}|\boldsymbol{M_2}]$, and since $\mathrm{rank}([\boldsymbol{M_1} \,|\, \boldsymbol{M_2}]) = r-j$, then for each choice of $[\boldsymbol{M_1} \,|\, \boldsymbol{M_1}]$ there are $2^{j(r-j)}$ possible $\boldsymbol{C}$ matrices. □

APPENDIX III
PROOFS OF LEMMAS AND THEOREMS OF SECTION VI

Proof of Lemma 6: The number of choices of the first p rows is $\prod_{j=0}^{t-1}(2^p-2^j)(2^n-2^j)/(2^t-2^j)$. Since the number of choices of the last $m-p$ rows does not depend on the structure of the first p rows, the specific matrix depicted in Fig. 13 can be considered. The rank of the $(m \times n)$ matrix is given by $t + \mathrm{rank}(\boldsymbol{M_2})$: then, $\mathrm{rank}(\boldsymbol{M_2}) = r-t$, and the number of $\boldsymbol{M_2}$ matrices is $\prod_{j=0}^{r-t-1}(2^{m-p}-2^j)(2^{n-t}-2^j)/(2^{r-t}-2^j)$. Since there are no constraints on the choice of $\boldsymbol{M_1}$, the number of $\boldsymbol{M_1}$ matrices for each choice of the first p rows and for each choice of $\boldsymbol{M_2}$ is $2^{t(m-p)}$. □

Proof of Lemma 7: $\tilde{F}(m,n,p,t,r)$ is equal to the total number of rank-r binary matrices such that the rank of the first p rows is t, i.e., $\tilde{N}(m,n,p,t,r)$, minus the number of such matrices with z zero columns, for $z = 1,\ldots,n-r$. There are $\binom{n}{z}$ choices for the z zero columns. Then, the number of rank-r $(m \times n)$ binary matrices such that the rank of the first

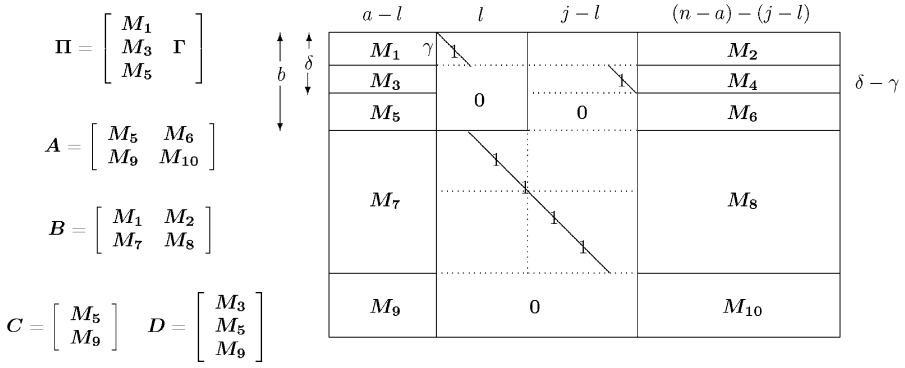

Fig. 17. Specific choice of the $(m \times n)$ matrix for the computation of $\hat{K}(m,n,a,b,t,u,r)$.

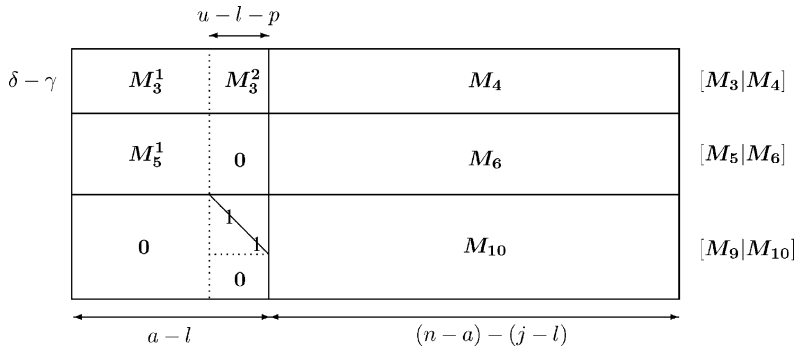

Fig. 18. Specific choice of the matrix $\boldsymbol{A}$ defined in Fig. 17 for the computation of the number of $[\boldsymbol{M_3}|\boldsymbol{M_4}]$ matrices.

p rows is t and with exactly z zero columns is expressed by $\binom{n}{z}\tilde{F}(m,n-z,p,t,r)$. □

Proof of Lemma 8: Let $\boldsymbol{M_1}$ be the submatrix composed of the first a columns of $\boldsymbol{G}$, and $\boldsymbol{M_2}$ be the submatrix composed of the last $n-a$ columns of $\boldsymbol{G}$. The number of $\boldsymbol{M_1}$ matrices is $\tilde{F}(m,a,b,t,u)$, expressed by Lemma 7. The number of $\boldsymbol{M_2}$ matrices is independent of the specific choice of $\boldsymbol{M_1}$. A convenient choice of $\boldsymbol{M_1}$ is depicted in Fig. 14, where $\boldsymbol{M_2}$ is partitioned into the three submatrices $\boldsymbol{M_2^1}, \boldsymbol{M_2^2}, \boldsymbol{M_2^3}$, and where the matrix $\boldsymbol{A}$ is defined. In order to have a total rank r, we must have $\mathrm{rank}(\boldsymbol{A}) = r-u$. Denoting by z the number of zero columns in $\boldsymbol{A}$, the number of $\boldsymbol{A}$ matrices is $\sum_{z=0}^{z_{\max}} \binom{n-a}{z} F(m-u, n-a-z, r-u)$, where $z_{\max} = (n-a)-(r-u)$ (because we need at least $r-u$ nonzero columns for $\boldsymbol{A}$). Since the overall $(m \times n)$ matrix must have no zero columns, the total number of choices for the z columns of $\boldsymbol{M_2^2}$, corresponding to the z zero columns of some choice of $\boldsymbol{A}$, is $T(u,z)$. Moreover, no constraint exists on the choice of the $n-a-z$ columns of $\boldsymbol{M_2^2}$ corresponding to the nonzero columns of $\boldsymbol{A}$. Then, this number is $2^{u(n-a-z)}$. □

Proof of Lemma 9: Since the rank of the $(m \times j)$ matrix is equal to j, all its columns must be linearly independent. Hence, the number of possible choices for the first l columns is $\tilde{F}(m,l,b,\gamma,l)$, with $\tilde{F}(\cdot)$ defined in Lemma 7. The number of possible choices for the last $j-l$ columns is independent of the specific choice of the first l columns. A convenient choice is depicted in Fig. 15, where the last $j-l$ columns are decomposed into the three submatrices $\boldsymbol{M_1}, \boldsymbol{M_2}$ and $\boldsymbol{M_3}$, and where the matrix $\boldsymbol{A}$ is defined. We prove next that, for each choice of the first l columns, the number of $\boldsymbol{A}$ matrices is $\tilde{F}(m-l, j-l, b-\gamma, \delta-\gamma, j-l)$ and, for each choice of the first l columns and $\boldsymbol{A}$ matrix, the number of $\boldsymbol{M_2}$ matrices is $2^{l(j-l)}$, thus leading to (28).

The total rank of the $(m \times j)$ matrix is equal to $l + \mathrm{rank}(\boldsymbol{A})$, so that $\mathrm{rank}(\boldsymbol{A}) = j-l$. Moreover, in order to have a rank δ for the first b rows, we must have $\mathrm{rank}(\boldsymbol{M_1}) = \delta-\gamma$. Since all the j columns must be linearly independent, $\boldsymbol{A}$ must have no zero columns. Then, the number of $\boldsymbol{A}$ matrices is $\tilde{F}(m-l, j-l, b-\gamma, \delta-\gamma, j-l)$. Since any choice is allowed for $\boldsymbol{M_2}$, the number of such submatrices is $2^{l(j-l)}$. □

Proof of Theorem 6: The number of desired binary matrices can be obtained as

$$\hat{K}(m,n,a,b,t,u,r) = \hat{M}(m,n,a,b,t,u,r) - \sum_{j=1}^{r-1} \hat{K}^{(j)}(m,n,a,b,t,u,r)$$

where $\hat{K}^{(j)}(m,n,a,b,t,u,r)$ is the number of rank-r $(m \times n)$ binary matrices without zero columns, such that the rank of the submatrix intersection of the first a columns and the first b rows is t, such that the rank of the first a columns is u and with exactly j independent columns. For each j, let l be the number of independent columns among the first a columns, and $j-l$ the number of independent columns among the last $n-a$ columns. Since the rank of the first a columns must be u, we must have $0 \le l \le u$. For each l, there are $\binom{a}{l}\binom{n-a}{j-l}$ possible positions for the j independent columns.

Let us assume that the j independent columns are the last l columns out of the first a columns, and the first $j-l$ columns out of the last $n-a$ columns, as shown in Fig. 16, where the matrices $\mathbf{\Gamma}$, $\mathbf{\Psi}$ and $\mathbf{\Delta}$ are defined. Denoting by γ the rank of the $(b \times l)$ matrix $\mathbf{\Gamma}$ and by δ the rank of the $(b \times j)$ matrix $\mathbf{\Delta}$, we have $0 \leq \gamma \leq \min\{b, l\}$ and $\gamma \leq \delta \leq \min\{b, j\}$.

For given γ and δ, the number choices for the j independent columns is $G(m, j, l, b, \gamma, \delta)$ as from Lemma 9. We can reason on the specific choice of the j independent columns depicted in Fig. 17, where the matrices $\mathbf{\Pi}$, $\boldsymbol{A}$, $\bar{\boldsymbol{B}}$, $\boldsymbol{C}$ and $\bar{\boldsymbol{D}}$ are defined. Let $\mathrm{rank}(\boldsymbol{M_5}) = p$. Since $\boldsymbol{M_5}$ is a $((b-\delta) \times (a-l))$ matrix, and since $\mathrm{rank}(\mathbf{\Pi}) = t$, we have $0 \leq p \leq \min\{b-\delta, a-l, t-\gamma\}$.

For each value of p, the number of $\boldsymbol{A}$ matrices is equal to $\hat{K}(m-j, n-j, a-l, b-\delta, p, u-l, r-j)$, as proved next. The $((m-j) \times (n-j))$ matrix $\boldsymbol{A}$ in Fig. 17 must have rank $r-j$, because the overall rank r is given by $j+\mathrm{rank}(\boldsymbol{A})$. It must have no zero columns due to the linear independence between the j independent columns and all the other columns. It must have no independent columns because, as the total rank is equal to $j + \mathrm{rank}(\boldsymbol{A})$, such columns would be independent columns for the overall $(m \times n)$ matrix which contradicts the hypothesis. The rank of the intersection between its first $a-l$ columns and $b-\delta$ rows is p. Finally, the rank of its first $a-l$ columns (i.e., $\mathrm{rank}(\boldsymbol{C})$ must be equal to $u-l$. This latter property can be proved as follows. By removing the j independent columns, we obtain a matrix with rank $r-j$, which is also the rank of $\boldsymbol{A}$. Then, each row in the matrix obtained by removing the j independent columns is a linear combination of the rows of $\boldsymbol{A}$. In particular, each row in $\boldsymbol{M_3}$ is a linear combination of the rows of $\boldsymbol{C}$, from which we obtain $\mathrm{rank}(\boldsymbol{D}) = \mathrm{rank}(\boldsymbol{C})$. Since the rank of the first a columns of the overall $(m \times n)$ matrix is u, it follows from Fig. 17 that $\mathrm{rank}(\boldsymbol{D}) = u-l$, that is $\mathrm{rank}(\boldsymbol{C}) = u-l$.

The number of rows of $\bar{\boldsymbol{B}}$ is $j-(\delta-\gamma)$, and the only condition on this matrix is that all its rows must be linear combinations of the rows of $\boldsymbol{A}$, whose rank is $r-j$. Then, the number of choices of $\bar{\boldsymbol{B}}$ is $2^{(r-j)(j-(\delta-\gamma))}$, that is independent of p.

The proof is completed by computing the number of $[\boldsymbol{M_3}|\boldsymbol{M_4}]$ matrices. We have $\mathrm{rank}(\mathbf{\Pi}) = t, \mathrm{rank}(\boldsymbol{M_5}) = p, \mathrm{rank}(\boldsymbol{C}) = u-l$ and any row in $[\boldsymbol{M_3}|\boldsymbol{M_4}]$ must be a linear combination of the rows of $\boldsymbol{A}$. Let us consider the specific choice of $\boldsymbol{A}$ depicted in Fig. 18, where $\mathrm{rank}(\boldsymbol{M_5^1}) = p$. The condition $\mathrm{rank}(\mathbf{\Pi}) = t$ is satisfied if and only if $\mathrm{rank}(\boldsymbol{M_3^2}) = t-\gamma-p$. Each row in $\boldsymbol{M_3^2}$ must be a linear combination of the $u-l-p$ rows of $\boldsymbol{A}$ corresponding to the $\bar{\boldsymbol{I}}_{u-l-p}$ matrix. All the possible $((\delta-\gamma) \times (u-l-p))$ $\boldsymbol{M_3^2}$ matrices can be generated with these vectors. Then, the number of $\boldsymbol{M_3^2}$ matrices is

$$\prod_{i=0}^{t-\gamma-p-1} \frac{(2^{\delta-\gamma}-2^i)(2^{u-l-p}-2^i)}{2^{t-\gamma-p}-2^i}.$$

Let us consider any specific choice of the matrix $\boldsymbol{M_3^2}$. Each row in $\boldsymbol{M_3^2}$ selects a specific linear combination of the $u-l-p$ rows of $\boldsymbol{A}$ that correspond to $\boldsymbol{I}_{u-l-p}$. The other rows define a matrix of rank $(r-j)-(u-l-p)$, so there are $2^{[(r-j)-(u-l-p)]}$ possible choices for each row of $[\boldsymbol{M_3^1} \mid \boldsymbol{M_4}]$. Since the total number of such rows is $\delta-\gamma$, the number of $[\boldsymbol{M_3^1}|\boldsymbol{M_4}]$ matrices is $2^{[(r-j)-(u-l-p)](\delta-\gamma)}$. □

ACKNOWLEDGMENT

The authors wish to thank Yige Wang for her feedback on this work and Gianluigi Liva for useful discussion.

REFERENCES

[1] R. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: M.I.T. Press, 1963.

[2] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 619–637, Feb. 2001.

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

[4] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[5] P. Oswald and M. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, no. 12, pp. 364–373, Dec. 2002.

[6] M. Shokrollahi, "New sequences of linear time erasure codes approaching the channel capacity," in *Proc. Int. Symp. Appl. Algebra, Algebraic Algorithms, Error Correcting Codes*, M. Fossorier, H. Imai, S. Lin, and A. Poli, Eds., Berlin, Germany, 1999, Lecture Notes in Computer Science, pp. 65–76.

[7] T. Richardson, "Error floors of LDPC codes," in *Proc. Forty-First Allerton Conf. Commun., Contr. Comput.*, Monticello, IL, Oct. 2003, pp. 1426–1435.

[8] M. Chiani and A. Ventura, "Design and performance evaluation of some high-rate irregular low-density parity-check codes," in *Proc. 2001 IEEE Global Telecommun. Conf.*, San Antonio, TX, Nov. 2001, vol. 2, pp. 990–994.

[9] A. Amraoui, A. Montanari, and R. Urbanke, "How to find good finite-length codes: From art towards science," *European Trans. Telecommun.*, vol. 18, no. 5, pp. 491–508, Aug. 2007.

[10] C. Di, R. Urbanke, and T. Richardson, "Weight distribution of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 4839–4855, Nov. 2006.

[11] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. 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, June 2002.

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

[13] M. Lentmaier and K. Zigangirov, "On generalized low-density parity-check codes based on Hamming component codes," *IEEE Commun. Lett.*, vol. 3, no. 8, pp. 248–250, Aug. 1999.

[14] J. Boutros, O. Pothier, and G. Zemor, "Generalized low density (Tanner) codes," in *Proc. 1999 IEEE Int. Conf. Commun.*, Vancouver, Canada, Jun. 1999, vol. 1, pp. 441–445.

[15] C. Measson and R. Urbanke, "Further analytic properties of EXIT-like curves and applications," in *Proc. 2003 IEEE Int. Symp. Inf. Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 266.

[16] I. Djordjevic, O. Milenkovic, and B. Vasic, "Generalized low-density parity-check codes for optical communication systems," *J. Lightw. Technol.*, vol. 23, no. 5, pp. 1939–1946, May 2005.

[17] M. Lentmaier, D. Truhachev, K. Zigangirov, and D. Costello, "An analysis of the block error probability performance of iterative decoding," *IEEE Trans. Inf. Theory*, vol. 51, no. 11, pp. 3834–3855, Nov. 2005.

[18] N. Miladinovic and M. Fossorier, "Generalized LDPC codes and generalized stopping sets," *IEEE Trans. Commun.*, vol. 56, no. 2, pp. 201–212, Feb. 2008.

[19] F. Kuo and L. Hanzo, "Symbol-flipping based decoding of generalized low-density parity-check codes over GF(q)," in *Proc. 2006 IEEE Wireless Commun. Netw. Conf.*, Las Vegas, NV, Apr. 2006, vol. 3, pp. 1207–1211.

[20] S. Abu-Surra, G. Liva, and W. Ryan, "Low-floor Tanner codes via Hamming-node or RSCC-node doping," in *Proc. of Int. Symp. Applied Algebra, Algebraic Algoritmhs, and Error Correcting Codes*, M. Fossorier, H. Imai, S. Lin, and A. Poli, Eds., Berlin, Germany, 2006, Lecture Notes in Computer Science, pp. 245–254.

[21] J. Chen and R. M. Tanner, "A hybrid coding scheme for the Gilbert-Elliott channel," *IEEE Trans. Commun.*, vol. 54, no. 10, pp. 1787–1796, Oct. 2006.

[22] G. Liva, W. Ryan, and M. Chiani, "Quasi-cyclic generalized LDPC codes with low error floors," *IEEE Trans. Commun.*, vol. 56, no. 1, pp. 49–57, Jan. 2008.
[23] G. Yue, L. Ping, and X. Wang, "Generalized low-density parity-check codes based on Hadamard constraints," *IEEE Trans. Inf. Theory*, vol. 53, no. 3, pp. 1058–1079, Mar. 2007.
[24] Y. Wang and M. Fossorier, "Doubly generalized LDPC codes over the AWGN channel," *IEEE Trans. Commun.*, vol. 57, no. 5, pp. 1312–1319, May 2009.
[25] Y. Wang and M. Fossorier, "EXIT chart analysis for doubly generalized LDPC codes," in *Proc. 2006 IEEE Global Telecommun. Conf.*, San Francisco, CA, Nov. 2006, pp. 1–6.
[26] S. Dolinar, "Design and iterative decoding of networks of many small codes," in *Proc. 2003 IEEE Int. Symp. Inf. Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 346.
[27] G. Zemor, "On expander codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 835–837, Feb. 2001.
[28] A. Barg and G. Zemor, "Distance properties of expander codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, pp. 78–90, Jan. 2006.
[29] S. Abu-Surra, G. Liva, and W. Ryan, "Design of generalized LDPC codes and their decoders," in *Proc. 2007 IEEE Commun. Theory Workshop*, Sedona, AZ, May 2007, p. 2.
[30] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.
[31] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.
[32] E. Sharon, A. Ashikhmin, and S. Litsyn, "Analysis of low-density parity-check codes based on EXIT functions," *IEEE Trans. Commun.*, vol. 54, no. 8, pp. 1407–1414, Aug. 2006.
[33] T. Helleseth, T. Kløve, and V. I. Levenshtein, "On the information function of an error-correcting code," *IEEE Trans. Inf. Theory*, vol. 43, no. 2, pp. 549–557, Mar. 1997.
[34] E. Paolini, M. Fossorier, and M. Chiani, "Doubly-generalized LDPC codes: Stability bound over the BEC," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 1027–1046, Mar. 2009.
[35] A. Barg, *Complexity Issues in Coding Theory, Handbook on Coding Theory, (Part I: Algebraic Coding)*. Amsterdam, The Netherlands: North Holland, 1998.
[36] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer-Verlag, 2005.
[37] *Handbook of Evolutionary Computation*, T. Back, D. Fogel, and Z. Michalewicz, Eds. Bristol, U.K.: IOP Publishing Ltd., 1997.
[38] K. Price and R. Storn, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimization*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[39] M. Shokrollahi and R. Storn, "Design of efficient erasure codes with differential evolution," in *Proc. IEEE Int. Symp. Inf. Theory*, Sorrento, Italy, Jun. 2000, p. 5.

Enrico Paolini (M'08) received the Dr. Ing. degree (with honors) in telecommunications engineering and the Ph.D. degree in telecommunications engineering from the University of Bologna, Italy, in 2003 and 2007, respectively.

While working towards the Ph.D. degree, he was Visiting Research Scholar at the University of Hawaii at Manoa. Currently, he holds a postdoctoral position at the Department of Electronics, Computer Science and Systems (DEIS) of the University of Bologna, Italy. His research interests include error-control coding (with emphasis on LDPC codes and their generalizations, iterative decoding algorithms, reduced-complexity maximum likelihood decoding for erasure channels), and distributed radar systems based on ultrawideband. In the field of error correcting codes, has been involved since 2004 in activities with the European Space Agency (ESA).

Dr. Paolini is member of the IEEE Communications Society and of the IEEE Information Theory Society.

Marc P. C. Fossorier (F'06) received the B.E. degree from the National Institute of Applied Sciences (INSA), Lyon, France, in 1987, and the M.S. and Ph.D. degrees in 1991 and 1994, respectively, all in electrical engineering.

His research interests include decoding techniques for linear codes, communication algorithms, and statistics.

Dr. Fossorier is a recipient of a 1998 NSF Career Development Award and became IEEE Fellow in 2006. He has served as Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY from 2003 to 2006, as Editor for the IEEE COMMUNICATIONS LETTERS from 1999 to 2008, as Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS from 1996 to 2003, and as Treasurer of the IEEE Information Theory Society from 1999 to 2003. From 2002 to 2008, he was also an Elected Member of the Board of Governors of the IEEE Information Theory Society which he served as Second Vice-President and First Vice-President. He was Co-Chairman of the 2007 International Symposium on Information Theory (ISIT), Program Co-Chairman for the 2000 International Symposium on Information Theory and Its Applications (ISITA) and Editor for the Proceedings of the 2006, 2003, and 1999 Symposiums on Applied Algebra, Algebraic Algorithms, and Error Correcting Codes (AAECC).

Marco Chiani (SM'02) was born in Rimini, Italy, in April 1964. He received the Dr. Ing. degree (*magna cum laude*) in electronic engineering and the Ph.D. degree in electronic and computer science from the University of Bologna, Italy, in 1989 and 1993, respectively.

He is a Full Professor at the II Engineering Faculty, University of Bologna, Italy, where he is the Chair in Telecommunication. During summer 2001, he was a Visiting Scientist at AT&T Research Laboratories in Middletown, NJ. He is a frequent visitor at the Massachusetts Institute of Technology (MIT), Cambridge, where he presently holds a Research Affiliate appointment. His research interests include wireless communication systems, MIMO systems, wireless multimedia, low-density parity-check codes (LDPCC) and UWB. He is leading the research unit of CNIT/University of Bologna on Joint Source and Channel Coding for wireless video and is a consultant to the European Space Agency (ESA-ESOC) for the design and evaluation of error correcting codes based on LDPCC for space CCSDS applications.

Dr. Chiani has chaired, organized sessions and served on the Technical Program Committees at several IEEE International Conferences. He was Co-Chair of the Wireless Communications Symposium at ICC 2004. In January 2006, he received the ICNEWS award *"For Fundamental Contributions to the Theory and Practice of Wireless Communications"*. He was the recipient of the *2008 IEEE ComSoc Radio Communications Committee Outstanding Service Award*. He is the past Chair (2002–2004) of the *Radio Communications Committee of the IEEE Communication Society* and past Editor of *Wireless Communication* (2000–2007) for the IEEE TRANSACTIONS ON COMMUNICATIONS.