Increasing the Lifetime of Flash Memories Using Multi-Dimensional Graph-Based Codes

Ahmed Hareedy, Rohith Kuditipudi, and Robert Calderbank
Electrical and Computer Engineering Department, Duke University, Durham, NC 27705 USA
ahmed.hareedy@duke.edu, rohith.kuditipudi@duke.edu, and robert.calderbank@duke.edu

***Abstract*—In order to meet the demands of data-hungry applications, data storage devices are required to be increasingly denser. Various sources of error appear with this increase in density. Multi-dimensional (MD) graph-based codes are capable of mitigating error sources like interference and channel non-uniformity in dense storage devices. Recently, a technique was proposed to enhance the performance of MD spatially-coupled codes that are based on circulants. The technique carefully relocates circulants to minimize the number of short cycles. However, cycles become more detrimental when they combine together to form more advanced objects, e.g., absorbing sets, including low-weight codewords. In this paper, we show how MD relocations can be exploited to minimize the number of detrimental objects in the graph of an MD code. Moreover, we demonstrate the savings in the number of relocation arrangements earned by focusing on objects rather than cycles. Our technique is applicable to a wide variety of one-dimensional (OD) codes. Simulation results reveal significant lifetime gains in practical Flash systems achieved by MD codes designed using our technique compared with OD codes having similar parameters.**

I. Introduction

The continuous and rapid growth in the density of modern storage devices brings many challenges. One of these challenges is an increase in the number of sources of data corruption in the system, which requires advanced error correcting codes to be applied. Because of their capacity-approaching performance and the degrees of freedom they offer in the code construction, graph-based codes, e.g., low-density parity-check (LDPC) codes, are applied in many data storage systems. Binary and non-binary graph-based codes are used in both Flash [1], [2] and magnetic recording [3] systems to significantly improve the performance.

Multi-dimensional (MD) graph-based codes are constructed by coupling different copies of a one-dimensional (OD) code to enhance the code properties. Because of the additional design flexibility offered by MD coupling, MD codes are capable of alleviating different types of interference and channel non-uniformity in modern storage systems. One example is mitigating inter-track interference in two-dimensional magnetic recording (TDMR) systems [3] through specific non-binary LDPC code constructions as in [4]. Various MD spatially-coupled (MD-SC) codes have been presented in the literature [5]–[8]. While these MD-SC codes demonstrated performance gains, they had limitations in the underlying OD codes and the topologies of the resulting MD codes.

Recently, the authors of [9] proposed a technique for a systematic construction of MD-SC codes that are based on circulants. Through carefully chosen relocations of circulants from the copies of the OD code to certain auxiliary matrices, they managed to significantly reduce the number of short cycles in the graph of the MD-SC code. While cycles are not preferred in graph-based codes, they become a lot more detrimental when they combine together to form absorbing sets (ASs), including low-weight codewords. ASs, not cycles, are the objects that dominate the error profile of graph-based codes in the error floor region [2], [10].

In this paper, we demonstrate how to use MD coupling to eliminate as many detrimental objects as possible from the graph of an MD code. The underlying OD codes we use can be structured or random, and can be block or SC codes. By deriving the fraction of relocation arrangements for different cases, we manifest the savings in relocation options achieved by operating on objects rather than cycles. Experimental results emphasizing the reduction in the multiplicity of detrimental objects are shown. Simulation results demonstrating ≈ 1200 (resp., 1800) program/erase cycles gain in the waterfall (resp., error floor) region over practical Flash channels compared with OD codes having similar length and rate are presented.

The rest of the paper is organized as follows. In Section II, MD graph-based codes are introduced. How ASs are removed via relocations is discussed in Section III. Next, the savings in relocation arrangements are derived in Section IV. In Section V, the code design algorithm and experimental results are presented. The paper is concluded in Section VI.

II. MD Graph-Based Codes

The technique we propose in this paper can be used to construct binary and non-binary codes. However, since the process of relocations affects only the code topology, we focus on the unlabeled graphs (all edge weights are set to 1) and binary matrices [2].

Define $\mathbf{H}_{\text{OD}}$ as the parity-check matrix of the underlying OD code, and $\mathbf{H}_{\text{MD}}$ as the parity-check matrix of the MD code. Recall the correspondence between the parity-check matrix and the graph of a code. Define $M-1$ auxiliary matrices, $\mathbf{X}_1$, $\mathbf{X}_2$, $\ldots$, $\mathbf{X}_{M-1}$ having the same dimensions as $\mathbf{H}_{\text{OD}}$. Then,

$$\mathbf{H}_{\text{MD}} \triangleq \begin{bmatrix} \mathbf{H}'_{\text{OD}} & \mathbf{X}_{M-1} & \mathbf{X}_{M-2} & \ldots & \mathbf{X}_2 & \mathbf{X}_1 \\ \mathbf{X}_1 & \mathbf{H}'_{\text{OD}} & \mathbf{X}_{M-1} & \ldots & \mathbf{X}_3 & \mathbf{X}_2 \\ \mathbf{X}_2 & \mathbf{X}_1 & \mathbf{H}'_{\text{OD}} & \ldots & \mathbf{X}_4 & \mathbf{X}_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{X}_{M-2} & \mathbf{X}_{M-3} & \mathbf{X}_{M-4} & \ldots & \mathbf{H}'_{\text{OD}} & \mathbf{X}_{M-1} \\ \mathbf{X}_{M-1} & \mathbf{X}_{M-2} & \mathbf{X}_{M-3} & \ldots & \mathbf{X}_1 & \mathbf{H}'_{\text{OD}} \end{bmatrix}, \tag{1}$$

where $\mathbf{H}_{\text{OD}} = \mathbf{H}'_{\text{OD}} + \sum_{\ell=1}^{M-1} \mathbf{X}_\ell$ (see also [9]). The graphs of the OD codes we use do not have any cycles of length 4. According to the construction of $\mathbf{H}_{\text{MD}}$, the data to be stored is separated into M chunks, and each chunk is stored in a track or a sector of the storage device. The matrix $\mathbf{H}_{\text{MD}}$ is constructed by coupling the M OD copies of $\mathbf{H}_{\text{OD}}$ via carefully relocating some of the non-zero (NZ) entries in these copies to auxiliary matrices in order to eliminate certain detrimental objects. Relocations are mathematically represented by an MD mapping as follows:

$$R : \{\mathcal{E}_{i,j}, \forall i, j\} \rightarrow \{0, 1, \ldots, M-1\}, \tag{2}$$

978-1-5386-6900-6/19/$31.00 ©2019 IEEE

Authorized licensed use limited to: Duke University. Downloaded on September 10,2020 at 21:22:30 UTC from IEEE Xplore. Restrictions apply.

where $\mathcal{E}_{i,j}$ is an NZ entry corresponding to an edge connecting check node (CN) i to variable node (VN) j in the graph of $\mathbf{H}_{\text{OD}}$. This mapping is explained as follows: $R(\mathcal{E}_{i,j}) = \ell > 0$ means that the NZ entry $\mathcal{E}_{i,j}$ is relocated from $\mathbf{H}_{\text{OD}}$ to $\mathbf{X}_\ell$ (M times) at the same position (i,j) it had in $\mathbf{H}_{\text{OD}}$, with $R(\mathcal{E}_{i,j}) = 0$ referring to the no-relocation case.

Here, M is a prime integer > 2, and both $\mathbf{H}_{\text{OD}}$ and $\mathbf{H}_{\text{MD}}$ have a fixed column weight, i.e., fixed VN degree, γ.

Define a cycle of length $2k$ in the graph of $\mathbf{H}_{\text{OD}}$ by the following set of NZ entries in $\mathbf{H}_{\text{OD}}$: $\{\mathcal{E}_{i_1,j_1}, \mathcal{E}_{i_2,j_2}, \ldots \mathcal{E}_{i_{2k},j_{2k}}\}$, such that two entries $\mathcal{E}_{i_w,j_w}$ and $\mathcal{E}_{i_{w+1},j_{w+1}}$, $1 \leq w \leq 2k$ and $\mathcal{E}_{i_{2k+1},j_{2k+1}} = \mathcal{E}_{i_1,j_1}$, are consecutive entries on the cycle. The authors of [9] proved that this cycle stays **active** after a relocation arrangement if and only if[1]:

$$\sum_{w=1}^{2k} (-1)^w R(\mathcal{E}_{i_w,j_w}) \equiv 0 \pmod{M}. \tag{3}$$

For a cycle of length $2k$ to stay active, its M copies must result in M cycles of length $2k$ in the graph of $\mathbf{H}_{\text{MD}}$. If (3) is not satisfied, the cycle becomes **inactive**, and its M copies result in a single cycle of length $2kM$. The result in [9] was for $M = 3$. However, this result generalizes to any prime M.

Under iterative decoding, the detrimental (error-prone) objects in the graph of a code are typically ASs. This is the case for additive white Gaussian noise (AWGN) [10], [12], Flash [2], [13], and magnetic recording [2] channels. Recall:

Definition 1. *Let $\mathcal{V}$ be a subset of VNs in the unlabeled graph of a code. Let $\mathcal{O}$ (resp., $\mathcal{T}$ and $\mathcal{H}$) be the set of degree-1 (resp., 2 and > 2) CNs connected to $\mathcal{V}$. This graphical configuration is an (a, d_1) unlabeled elementary absorbing set (UAS) if $|\mathcal{V}| = a$, $|\mathcal{O}| = d_1$, $|\mathcal{H}| = 0$, and each VN in $\mathcal{V}$ is connected to strictly more neighbors in $\mathcal{T}$ than in $\mathcal{O}$.*

Remark 1. *Many non-elementary absorbing sets appearing in the error profile of non-binary graph-based codes over practical Flash channels have underlying unlabeled elementary configurations [2].*

We study UASs having connected subgraphs. A $(4,2)$ UAS in a code with $\gamma = 3$ and a $(4,4)$ UAS in a code with $\gamma = 4$ are shown in Fig. 1. Circles (resp., grey and white squares) represent VNs (resp., degree-1 and degree-2 CNs). We will investigate how to perform relocations to minimize the number of UASs in an MD code to enhance its performance.

III. Removing ASs Through Relocations

An (a, d_1) UAS has the following number of degree-2 CNs:

$$d_2 = \frac{1}{2}(a\gamma - d_1). \tag{4}$$

We now revisit the concept of **basic cycles**, which generalizes the concept of fundamental cycles first introduced in [12] for non-binary codes, to represent a UAS.

Definition 2. *A cycle basis $\mathcal{B}_{\text{c}}$ of an (a, d_1) UAS is a minimum-cardinality set of cycles using disjunctive unions of which, each cycle in the UAS can be obtained. We call the cycles in $\mathcal{B}_{\text{c}}$ basic cycles.*

Denote a Galois field of size q as GF(q). Since our graphs are unlabeled (no weights), span($\mathcal{B}_{\text{c}}$) can be represented by a vector space over GF(2), with its vectors being of size $n_{\text{e}} = 2d_2$ and their elements are also in GF(2). There are $n_{\text{f}} = |\mathcal{B}_{\text{c}}|$ basic cycles. From graph theory principles, this number is computed by subtracting the number of degree-2 CNs, each represented by the pair of edges adjacent to it, comprising the tree spanning all VNs from the total number of degree-2 CNs. Consequently,

$$\begin{aligned} n_{\text{f}} &= \frac{1}{2}\left(n_{\text{e}} - 2(a-1)\right) = d_2 - a + 1 \\ &= \frac{1}{2}\left(a(\gamma-2) - d_1 + 2\right), \end{aligned} \tag{5}$$

where the last equality is obtained using (4). Without loss of generality, in this paper, we always select the basic cycles in $\mathcal{B}_{\text{c}}$ to be of the smallest lengths for simplicity.

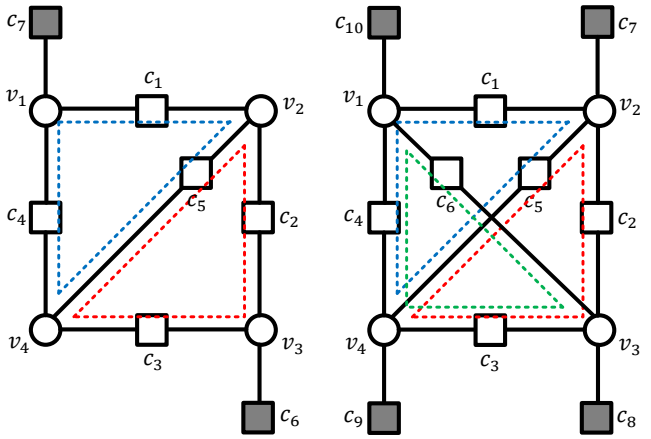

Fig. 1. Left panel: a $(4,2)$ UAS, $\gamma = 3$. Right panel: a $(4,4)$ UAS, $\gamma = 4$. Basic cycles are shown in dotted lines.

Example 1. *Consider the $(4,2)$ UAS, $\gamma = 3$, in Fig. 1. From (5), the number of basic cycles is $n_{\text{f}} = \frac{1}{2}(4(3-2) - 2 + 2) = 2$. We select the two cycles in dotted blue and dotted red shown in the figure to be the elements of $\mathcal{B}_{\text{c}}$. A cycle in span($\mathcal{B}_{\text{c}}$) can be written as $\left[e_{c_1,v_1}\ e_{c_1,v_2}\ e_{c_2,v_2}\ e_{c_2,v_3}\ e_{c_3,v_3}\ e_{c_3,v_4}\ e_{c_4,v_4}\ e_{c_4,v_1}\ e_{c_5,v_2}\ e_{c_5,v_4}\right]$, where $e_{i_w,j_w} = \mathbb{1}(\mathcal{E}_{i_w,j_w})$ is an indicator function of the existence of the NZ entry $\mathcal{E}_{i_w,j_w}$. Thus, the dotted blue and dotted red basic cycles are $[1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1]$ and $[0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1]$, respectively. Adding the vectors of the two basic cycles over GF(2) gives $[1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0]$, which is the vector of the only remaining cycle in the UAS.*

Given the number of basic cycles in an (a, d_1) UAS, we now introduce useful bounds on the total number of cycles.

Lemma 1. *The total number of cycles, n_{c}, in an (a, d_1) UAS having n_{f} basic cycles is bounded as follows:*

$$\frac{1}{2} n_{\text{f}}(n_{\text{f}} + 1) \leq n_{\text{c}} \leq 2^{n_{\text{f}}} - 1. \tag{6}$$

Proof: See [14], which is the long version of the paper. ■

Example 2. *The upper and the lower bounds are the same for the $(4,2)$ UAS, $\gamma = 3$, in Fig. 1. Since $n_{\text{f}} = 2$, $n_{\text{c}} = \frac{1}{2}2(2+1) = 3$ from (6). On the contrary, only the upper bound is achieved for the $(4,4)$ UAS, $\gamma = 4$, in Fig. 1. Since $n_{\text{f}} = 3$ from (5), $n_{\text{c}} = 2^3 - 1 = 7$ from (6).*

We are now ready to introduce the condition under which a UAS stays **active** after a relocation arrangement. For an (a, d_1)

[1]This condition bares similarity to the condition in [11] for protograph lifting. In fact, some of the results in this paper are applicable to the procedures of lifting and non-binary labeling.

Authorized licensed use limited to: Duke University. Downloaded on September 10,2020 at 21:22:30 UTC from IEEE Xplore. Restrictions apply.

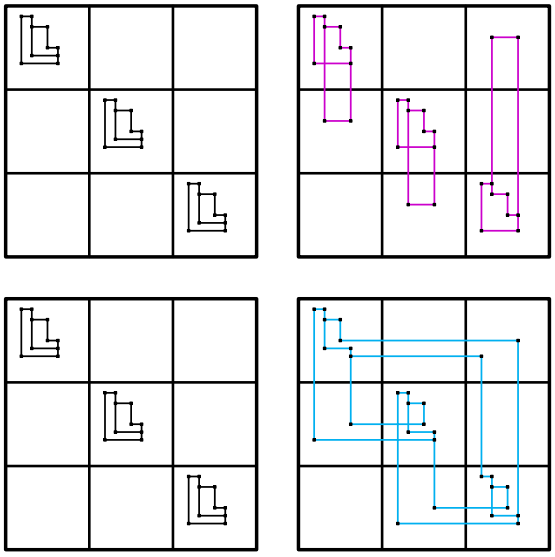

Fig. 2. Upper panel: Arrangement 1 is keeping three instances of the $(4,2)$ UAS in $\mathbf{H}_{\text{MD}}$. Lower panel: Arrangement 2 is removing the three copies of the $(4,2)$ UAS from $\mathbf{H}_{\text{MD}}$. VNs of the $(4,2)$ UAS, which are columns in the matrix, are ordered from left to right as v_1, v_2, v_3, and v_4 (see Fig. 1).

UAS to stay active, its M copies in the graphs of $\mathbf{H}_{\text{OD}}$ copies must result in M (a,d_1) UASs in the graph of $\mathbf{H}_{\text{MD}}$.

Theorem 1. *The necessary and sufficient condition for an (a,d_1) UAS to stay active after a relocation arrangement is that (3) is satisfied for all the n_{f} basic cycles in a cycle basis $\mathcal{B}_{\text{c}}$ of the UAS. Otherwise, the UAS becomes inactive, and the Ma VNs of its M copies form an (Ma,Md_1) object.*

Proof: See [14]. ∎

If the UAS becomes inactive after relocations, its M copies are **removed** from the graph of $\mathbf{H}_{\text{MD}}$. Depending on certain factors, including which cycles in the (a,d_1) UAS become inactive after relocations, different, possibly non-isomorphic, (Ma,Md_1) configurations can be generated if the UAS is inactive. On a smaller scale, the M copies of the (a,d_1) UAS result in multiple $(a,d_1+2\beta)$ objects, $\beta>0$, in this case.

Example 3. *Consider an instance of the $(4,2)$ UAS, $\gamma=3$, in Fig. 1, which exists in $\mathbf{H}_{\text{OD}}$, and let $M=3$ for $\mathbf{H}_{\text{MD}}$. The three copies of the UAS in $\mathbf{H}_{\text{MD}}$ are shown in the left panel of Fig. 2 (degree-1 CNs are not shown). We check the following two relocation arrangements:*

Arrangement 1: $R(\mathcal{E}_{c_5,v_2})=R(\mathcal{E}_{c_5,v_4})=1$, while all the remaining NZ entries of the UAS are not relocated. Here, (3) is satisfied for both the dotted blue and the dotted red basic cycles. Thus, the $(4,2)$ UAS stays active, which is shown in the upper panel of Fig. 2.

Arrangement 2: $R(\mathcal{E}_{c_3,v_4})=R(\mathcal{E}_{c_4,v_1})=1$, while all the remaining NZ entries of the UAS are not relocated. Here, (3) is not satisfied for either basic cycle. Thus, the $(4,2)$ UAS becomes inactive, which is shown in the lower panel of Fig. 2. How the three copies of the $(4,2)$ UAS result in a $(12,6)$ object after relocations is demonstrated in [14, Fig. 3].

IV. Savings in Relocation Options

Targeting UASs instead of the cycles comprising them not only makes the focus in the code design on the more detrimental objects, but also achieves significant savings in the degrees of freedom offered by relocation arrangements. These savings are reflected in performance gains. Here, we demonstrate these savings. In the following results, fractions are out of all possible relocation arrangements. Let $(x)^+=\max\{x,0\}$.

Lemma 2 discusses the relocation arrangements in case the focus is on removing short cycles from the graph of $\mathbf{H}_{\text{MD}}$.

Lemma 2. *The fraction of relocation arrangements for an (a,d_1) UAS under which all the basic cycles in a cycle basis $\mathcal{B}_{\text{c}}$ of the UAS become inactive is given by:*

$$F_{\text{nof}}=\left(\frac{M-1}{M}\right)^{n_{\text{f}}}. \tag{7}$$

Moreover, the fraction of relocation arrangements for an (a,d_1) UAS under which all its cycles become inactive is upper-bounded as follows:

$$F_{\text{noc}}\le\prod_{\delta=1}^{n_{\text{f}}}\left(\frac{M-\delta}{M}\right)^{+}. \tag{8}$$

Proof: See [14]. ∎

Theorem 2 discusses the relocation arrangements in case the focus is on removing UASs from the graph of $\mathbf{H}_{\text{MD}}$.

Define $\mathcal{F}_i$ as the set of CNs in basic cycle i, and $\mathcal{I}_{i,j}\triangleq\mathcal{F}_i\cap\mathcal{F}_j$. Moreover,

$$\mathcal{I}_i^{\text{tot}}\triangleq\bigcup_j(\mathcal{I}_{i,j}),\ \text{and}\ \mathcal{D}_i\triangleq\mathcal{F}_i\setminus\mathcal{I}_i^{\text{tot}}. \tag{9}$$

Let D_i be the unordered group comprising the CNs of $\mathcal{D}_i$. Then, we define the following set:

$$\mathcal{L}_1\triangleq\{D_i,\forall i\mid\mathcal{D}_i\neq\varnothing\}. \tag{10}$$

Let $I_{i,j}$ be the unordered group comprising the CNs of $\mathcal{I}_{i,j}$. Then, we define the following set:

$$\mathcal{L}_2\triangleq\{I_{i,j},\forall i,j\mid\mathcal{I}_{i,j}\neq\varnothing\}. \tag{11}$$

Theorem 2. *The fraction of relocation arrangements for an (a,d_1) UAS under which the UAS becomes inactive is given by:*

$$F_{\text{nou}}=1-\frac{1}{M^{n_{\text{f}}}}. \tag{12}$$

Moreover, the fraction of relocation arrangements for an (a,d_1) UAS under which the M copies of the UAS result in at least M $(a,d_1+2\beta)$ objects, with $\beta>1$, is given by:

$$F_{\text{not}}=1-\frac{1}{M^{n_{\text{f}}}}-\left[|\mathcal{L}_1|+|\mathcal{L}_2|\right]\frac{(M-1)}{M^{n_{\text{f}}}}. \tag{13}$$

Proof: See [14]. ∎

On the level of an object, the percentage saving in relocation arrangements achieved by focusing on the UAS instead of focusing on all its cycles is given by:

$$\begin{aligned}S_1&=[F_{\text{nou}}-\text{bound}(F_{\text{noc}})]\cdot100\%\\&=\left[1-\frac{1}{M^{n_{\text{f}}}}-\prod_{\delta=1}^{n_{\text{f}}}\left(\frac{M-\delta}{M}\right)^{+}\right]\cdot100\%.\end{aligned} \tag{14}$$

Another realization of the saving is given in [14].

Example 4. *Consider the $(4,2)$ UAS, $\gamma=3$, in Fig 1. Let $M=5$. From Example 1, $n_{\text{f}}=2$. Thus, from (7) and (8),*

$$F_{\text{nof}}=\left(\frac{4}{5}\right)^2=\frac{16}{25},\ F_{\text{noc}}\le\prod_{\delta=1}^{2}\left(\frac{5-\delta}{5}\right)^{+}=\frac{12}{25}.$$

The two basic cycles here have $\mathcal{F}_1=\{c_1,c_4,c_5\}$ and $\mathcal{F}_2=\{c_2,c_3,c_5\}$. Consequently, we get $\mathcal{I}_{1,2}=\{c_5\}$, yielding $\mathcal{I}_1^{\text{tot}}=\mathcal{I}_2^{\text{tot}}=\{c_5\}$. From (9), $\mathcal{D}_1=\{c_1,c_4\}$ and $\mathcal{D}_2=\{c_2,c_3\}$.

Authorized licensed use limited to: Duke University. Downloaded on September 10,2020 at 21:22:30 UTC from IEEE Xplore. Restrictions apply.

Thus, from (10) and (11), we get $\mathcal{L}_1 = \{(c_1, c_4), (c_2, c_3)\}$ *and* $\mathcal{L}_2 = \{(c_5)\}$. *From (12) and (13),*

$$F_{\text{nou}} = 1 - \frac{1}{5^2} = \frac{24}{25},\ F_{\text{not}} = 1 - \frac{1}{5^2} - [3]\frac{4}{5^2} = \frac{12}{25}.$$

Now, we are ready to calculate the saving in relocation arrangements from (14) as follows:

$$S_1 = \left[\frac{24}{25} - \frac{12}{25}\right] \cdot 100\% = 48\%,$$

which is a significant saving.

Now, we briefly introduce a special case of interest.

Definition 3. *Let* $a_{\min}$ *be the minimum UAS size in the OD code. An* (a, d_1) *UAS,* $a < Ma_{\min}$, *is said to be non-regenerable if it cannot be produced from* (a, d_1^-) *UASs,* $\psi^- < \psi$, *under any relocation arrangement. Furthermore, an* (a, d_1) *UAS is said to be stand-alone if an instance of this UAS cannot share any cycles with another instance of it.*

For example, $(a, 0)$ UASs are non-regenerable and stand-alone. Additionally, UASs with $d_2 = \binom{a}{2}$ are non-regenerable.

For non-regenerable, stand-alone UASs, the savings in relocation arrangements can be generalized over the entire graph of the MD code. More intriguingly, under random relocations, the average number of instances of an (a, d_1) non-regenerable, stand-alone UAS in the graph of the MD code is given by:

$$\overline{A}_{\text{MD}} = A_{\text{OD}} F_0 M, \tag{15}$$

where A_{OD} is the number of instances in the OD code, and $F_0 = \frac{1}{M^{n_{\text{f}}}}$, which is helpful in code optimization.

V. Algorithm and Experimental Results

Guided by the proposed theoretical results, Algorithm 1 minimizes the number of instances of a specific (a, d_1) UAS, $a < Ma_{\min}$, in the graph of the MD code via relocations. This specific (a, d_1) UAS/AS can either be the most dominant object in the error profile of the OD code or a common substructure that exists in the most dominant UASs in the OD code (over the specific channel of interest). Because of their faster encoding and decoding, we focus on circulant-based codes in this section. Since operating on circulants is significantly faster than operating on entries, Algorithm 1 relocates NZ circulants, not NZ entries (see [9]). The algorithm can be easily changed to relocate NZ entries for codes that are not structured.

We say that an (a, d_1) UAS instance **involves** a circulant if the instance has at least one NZ entry corresponding to an edge adjacent to a degree-2 CN inside the circulant. Moreover, the set of relocation decisions is $\mathcal{X} = \{0, 1, \ldots, M-1\}$. The value $\xi \in \mathcal{X}$, $\xi > 0$ (resp., $\xi = 0$), refers to the decision "relocate to $\mathbf{X}_\xi$" (resp., "no relocation"). Note that Step 16 of Algorithm 1 aims to balance the number of NZ circulants (similar sparsity levels) across all auxiliary matrices in addition to its main objective, which is removing (a, d_1) UAS instances.

Next, we discuss the experimental results. The Flash channel used in this section is a practical, asymmetric Flash channel, which is the normal-Laplace mixture (NLM) Flash channel [1]. In the NLM channel, the threshold voltage distribution of sub-20nm multi-level cell (MLC) Flash memories is carefully modeled. The four levels are modeled as different NLM distributions, incorporating several sources of error due to wear-out effects, e.g., programming errors, thereby resulting in significant asymmetry. Furthermore, the authors provided accurate fitting results of their model for program/erase (P/E) cycles up to 10 times the manufacturer's endurance specification (up to 30000 P/E cycles). We implemented the NLM channel based on the parameters described in [1]. Here, we use 3 reads, and the sector size is 512 bytes. For decoding, we use a fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder (see also [2]).

Algorithm 1 Designing High Performance MD Codes

1: **Inputs:** $\mathbf{H}_{\text{OD}}$, M, and the (a, d_1) UAS configuration.
2: Initially, set $\mathbf{X}_1 = \mathbf{X}_2 = \cdots = \mathbf{X}_{M-1} = \mathbf{0}$, $\mathbf{H}'_{\text{OD}} = \mathbf{H}_{\text{OD}}$, and $R(\mathcal{E}_{i,j}) = 0$, $\forall i, j$.
3: Locate all instances of the (a, d_1) UAS in $\mathbf{H}_{\text{OD}}$.
4: Mark all the instances located in Step 3 as active.
5: Determine the number of active (a, d_1) UAS instances involving each NZ circulant in $\mathbf{H}_{\text{OD}}$.
6: Select the circulant $\mathcal{C}$ with the maximum number from Step 5 s.t. $R(\mathcal{E}_{i_t,j_t}) = 0$, where $\mathcal{E}_{i_t,j_t}$ is an NZ entry in $\mathcal{C}$.
7: Whether they are active or not, specify all (a, d_1) UAS instances in $\mathbf{H}_{\text{OD}}$ involving $\mathcal{C}$.
8: **for** each of the instances from Step 7 **do**
9: Specify a cycle basis $\mathcal{B}_{\text{c}}$ of the instance.
10: The instance votes for the subset of decisions in $\mathcal{X}$ that make at least one of its basic cycles in $\mathcal{B}_{\text{c}}$ inactive.
11: **end for**
12: Tally the votes, and find the subset $\mathcal{W}$ of NZ decisions in $\mathcal{X}$ with the highest number of votes.
13: **if** $\mathcal{W} = \varnothing$ **then** *(no relocation)*
14: Go to Step 22.
15: **end if**
16: Relocate $\mathcal{C}$ to the auxiliary matrix $\mathbf{X}_{\xi_r}$, $\xi_r \in \mathcal{W}$, with the least number of NZ circulants.
17: Set $R(\mathcal{E}_{i,j}) = \xi_r$ for all NZ entries in $\mathcal{C}$.
18: Update the list of active/inactive (a, d_1) instances based on their basic cycles and Theorem 1.
19: **if** the number of active (a, d_1) instances is > 0 **then**
20: Go to Step 5.
21: **end if**
22: Construct $\mathbf{H}_{\text{MD}}$ according to (1).
23: **Output:** The parity-check matrix of the MD code, $\mathbf{H}_{\text{MD}}$.

We use three OD codes in this section. The SC (resp., block) code is designed according to [13] (resp., [2]). OD Code 1 is an SC code defined over GF(4), which has $\gamma = 3$, maximum row weight $= 19$, circulant size $= 19$, memory $= 1$, and coupling length $= 7$. Thus, OD Code 1 has block length $= 5054$ bits and rate ≈ 0.82. OD Code 2 is a block code defined over GF(2), which has $\gamma = 4$, row weight $= 40$, and circulant size $= 53$. OD Code 2 has block length $= 4240$ bits and rate ≈ 0.90. OD Code 3 is an SC code that is designed exactly as OD Code 1, but with coupling length $= 21$. Thus, OD Code 3 has block length $= 15162$ bits and rate ≈ 0.83.

From our simulations, the error profile in the error floor region of OD Code 1 (resp., OD Code 2) when simulated over the NLM (resp., AWGN) channel is dominated by the $(4, 2)$ non-binary AS (resp., the $(4, 4)$ and the $(6, 2)$ UASs). The overwhelming majority of the $(6, 2)$ UAS instances found in the error profile of OD Code 2 have the same configuration, which has the $(4, 4)$ UAS as a substructure. Observe that OD Code 1 and OD Code 2 are the underlying OD codes of the MD codes used in this section.

Remark 2. *The objects of interest in other codes and over other channels can be more sophisticated, e.g., the* $(6, 0)$ *and*

Authorized licensed use limited to: Duke University. Downloaded on September 10,2020 at 21:22:30 UTC from IEEE Xplore. Restrictions apply.

TABLE I
EFFECT OF CAREFULLY CHOSEN MD RELOCATIONS ON THE NUMBER OF $(4,2)$ UAS ($\gamma = 3$) AND $(4,4)$ UAS ($\gamma = 4$) INSTANCES.

MD coupling technique	Number of $(4,2)$ UAS instances	Number of $(4,4)$ UAS instances
No MD coupling	4218	3392
Algorithm 1	0	0

the $(8,0)$ *UASs,* $\gamma = 3$*, in addition to the* $(6,6)$ *and the* $(8,2)$ *UASs,* $\gamma = 4$*. See [2] for more details.*

As for the MD codes, MD Code 1 (resp., MD Code 2) is designed for practical Flash (resp., AWGN) channels. According to the analysis above, MD Code 1 (resp., MD Code 2), with $M = 3$, is designed from OD Code 1 (resp., OD Code 2) using Algorithm 1 by removing as many $(4,2)$ UAS (resp., $(4,4)$ UAS) instances as possible in the MD code via relocations. MD Code 1 has block length $= 15162$ bits and rate ≈ 0.82, which is similar to OD Code 3 (the long OD SC code). MD Code 2 has block length $= 12720$ bits and rate ≈ 0.90.

Table I demonstrates the reduction in the number of detrimental objects achieved by Algorithm 1. The no-MD-coupling case refers to the case when $\mathbf{H}_{\text{MD}}$ is constructed by putting three copies of $\mathbf{H}_{\text{OD}}$ in the block diagonal and zeros elsewhere. Table I shows that, and with only about 7% (resp., 4.5%) of the circulants relocated out of the OD copies to construct MD Code 1 (resp., MD Code 2), Algorithm 1 removes all the $(4,2)$ UAS (resp., $(4,4)$ UAS) instances. These relatively small percentages of relocated circulants exemplify the savings in relocation arrangements (discussed in Section IV) in the MD code design, making it possible to relocate more circulants in order to remove other detrimental objects.

Here, RBER is the raw bit error rate, which is the number of uncoded data bits in error divided by the total number of uncoded data bits read [2]. UBER is the uncorrectable bit error rate, which is the frame error rate (FER) after error correction is applied divided by the sector size in bits [2].

Fig. 3 demonstrates the performance gains achieved by an MD code constructed using Algorithm 1, which is MD Code 1, compared with an OD code of similar length and rate, which is OD Code 3, over the practical NLM Flash channel. In particular, at UBER $\approx 10^{-7}$ (resp., $\approx 10^{-9}$) in the waterfall (resp., error floor) region, the RBER gain of MD Code 1 marked in red translates to a gain of about 1200 (resp., 1800) P/E cycles. Additionally, even the threshold of MD Code 1 is indeed better than that of OD Code 3. These gains in the number of P/E cycles are associated with an increase in the lifetime of the Flash device.

Remark 3. *While we focus here on practical Flash channels in the simulations, performance gains are also achievable via the proposed technique on other channels.*

VI. CONCLUSION

We introduced necessary and sufficient conditions for a UAS to stay active or become inactive, i.e., be removed, after a relocation arrangement. We derived the savings in relocation options achieved by focusing on UASs instead of cycles in the MD code design procedure. Examples demonstrating the significance of these savings were introduced for famous UAS configurations. We presented an algorithm to design high performance MD codes by removing detrimental UASs via relocations. Using this algorithm, codes free of specific UASs were designed and simulated. Gains of up to about 1800 P/E cycles were achieved via our MD codes compared with OD codes of similar parameters over a practical Flash channel.

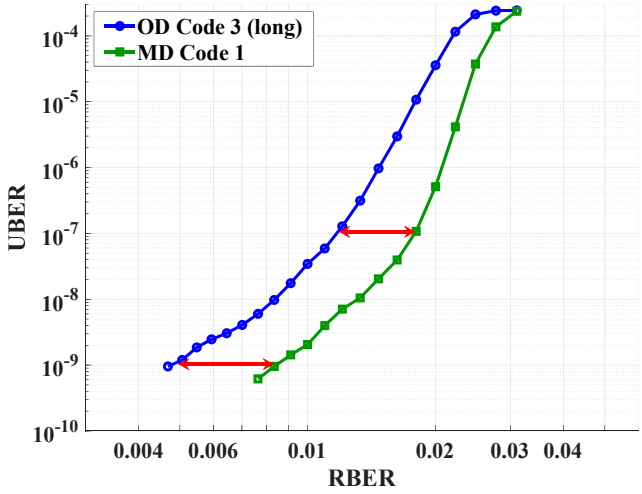

Fig. 3. UBER versus RBER curves over the NLM Flash channel for OD and MD codes of similar parameters.

ACKNOWLEDGMENT

This research was supported in part by NSF under grant CCF 1717602.

REFERENCES

[1] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20nm NAND flash memory," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Austin, TX, USA, Dec. 2014, pp. 2351–2356.

[2] A. Hareedy, C. Lanka, N. Guo, and L. Dolecek, "A combinatorial methodology for optimizing non-binary graph-based codes: theoretical analysis and applications in data storage," *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 2128–2154, Apr. 2019.

[3] S. Srinivasa, Y. Chen, and S. Dahandeh, "A communication-theoretic framework for 2-DMR channel modeling: performance evaluation of coding and signal processing methods," *IEEE Trans. Magn.*, vol. 50, no. 3, pp. 6–12, Mar. 2014.

[4] P. Chen, C. Kui, L. Kong, Z. Chen, M. Zhang, "Non-binary protograph-based LDPC codes for 2-D-ISI magnetic recording channels," *IEEE Trans. Magn.*, vol. 53, no. 11, Nov. 2017, Art. no. 8108905.

[5] D. Truhachev, D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "New codes on graphs constructed by connecting spatially coupled chains," in *Proc. Inf. Theory and App. Workshop (ITA)*, Feb. 2012, pp. 392–397.

[6] R. Ohashi, K. Kasai, and K. Takeuchi, "Multi-dimensional spatially-coupled codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2013, pp. 2448–2452.

[7] L. Schmalen and K. Mahdaviani, "Laterally connected spatially coupled code chains for transmission over unstable parallel channels," in *Proc. Int. Symp. Turbo Codes Iterative Inf. Processing (ISTC)*, Aug. 2014, pp. 77–81.

[8] Y. Liu, Y. Li, and Y. Chi, "Spatially coupled LDPC codes constructed by parallelly connecting multiple chains," *IEEE Commun. Letters*, vol. 19, no. 9, pp. 1472–1475, Sep. 2015.

[9] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Multi-dimensional spatially-coupled code design through informed relocation of circulants," in *Proc. 56th Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, USA, Oct. 2018, pp. 695–701.

[10] L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright, and B. Nikolic, "Analysis of absorbing sets and fully absorbing sets of array-based LDPC codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 181–201, Jan. 2010.

[11] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[12] B. Amiri, J. Kliewer, and L. Dolecek, "Analysis and enumeration of absorbing sets for non-binary graph-based codes," *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 398–409, Feb. 2014.

[13] A. Hareedy, H. Esfahanizadeh, and L. Dolecek, "High performance non-binary spatially-coupled codes for Flash memories," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017, pp. 229–233.

[14] A. Hareedy, R. Kuditipudi, and R. Calderbank, "Minimizing the number of detrimental objects in multi-dimensional graph-based codes," Apr. 2019. [Online]. Available: https://arxiv.org/abs/1904.03844

Authorized licensed use limited to: Duke University. Downloaded on September 10,2020 at 21:22:30 UTC from IEEE Xplore. Restrictions apply.