Rank-Modulation Rewriting Codes for Flash Memories

Eyal En Gad[*], **Eitan Yaakobi**[*], **Anxiao (Andrew) Jiang**[†] and **Jehoshua Bruck**[*]
[*]Electrical Engineering, California Institute of Technology, Pasadena, CA 91125.
[†]Computer Science and Engineering, Texas A&M University, College Station, TX 77843.
[*]{*eengad,yaakobi,bruck*}@*caltech.edu*, [†]*ajiang@cse.tamu.edu*

Abstract—**Current flash memory technology is focused on cost minimization of the stored capacity. However, the resulting approach supports a relatively small number of write-erase cycles. This technology is effective for consumer devices (smartphones and cameras) where the number of write-erase cycles is small, however, it is not economical for enterprise storage systems that require a large number of lifetime writes.**

Our proposed approach for alleviating this problem consists of the efficient integration of two key ideas: (i) improving reliability and endurance by representing the information using relative values via the rank modulation scheme and (ii) increasing the overall (lifetime) capacity of the flash device via rewriting codes, namely, performing multiple writes per cell before erasure.

We propose a new scheme that combines rank-modulation with rewriting. The key benefits of the new scheme include: (i) the ability to store close to 2 bits per cell on each write, and rewrite the memory close to q times, where q is the number of levels in each cell, and (ii) efficient encoding and decoding algorithms that use the recently proposed polar WOM codes.

I. Introduction

The application of the *rank-modulation* scheme for flash memories was proposed by Jiang et al. in [10]. The main idea of this modulation scheme is to represent the information by the *relative* levels of the flash memory cells, rather than by their absolute levels. Given a set of flash cells with distinct levels, the levels induce a *permutation*, which represents the stored data. The motivation for the scheme comes from the physical and architectural properties of flash memories. While injecting charge into a flash cell is a simple operation, removing it can by done only by the removal of the *entire* charge from a large block of cells, a process called *block erasure*. In conventional Multi-Level Cell (MLC) flash systems, the information is represented by the quantization of the cells' levels. Since the charge injection operation is a noisy process, it is often done iteratively, in order to avoid undesired block erasures in case of overshoots. It was suggested in [10] that the rank-modulation scheme speeds up data writing by eliminating the over-shooting problem in flash memories. In addition, it also increases the data retention by mitigating the effect of charge leakage. A hardware implementation of the scheme was recently designed to demonstrate those properties [12].

The work on rank modulation coding for flash memories paved the way for additional results in this area. First, error-correcting codes in the rank modulation setup attracted a lot of attention; see e.g. [2], [7], [11], [16]. In addition, other variations of rank modulation were proposed and studied, such as [6], [17].

In this work we focus on the notion of *rewriting codes*, that were proposed for the rank-modulation scheme in [10], in order to *reuse* the memory between block erasures. Since block erasures are slow, power consuming and are reducing the device reliability, it is desirable to minimize their usage. This is especially important in applications that require a large number of writes, such as enterprise storage systems. In order to minimize block erasures, the proposed approach is to rewrite the memory without erasing it, by injecting charge to the cells such that they induce a desired new permutation, and thus represent a new user message. After a number of rewriting cycles, the cells reach their maximal level, and block erasure is unavoidable. The aim of rewriting codes is to maximize the number of writes between block erasures.

In rank-modulation, each cell has a certain *rank*, according to its relative level in the permutation. Depending on the resolution of charge detection and the noise magnitude, a certain gap is needed between cells of adjacent rank, to avoid errors. Therefore, it was proposed in [4] to use a discrete model for the design and analysis of rewriting codes, despite the fact that the information is only based on the relative *analog* levels of the cells. The approach taken in [4] is to focus, in every rewrite, on the difference between the levels of the *top* cell in the permutation, before and after the rewrite. This difference is defined as the *cost* of rewrite. The reason for this focus is that writing with high cost gets the memory closer to the point where block erasure is required. Under this model, the goal of this work is to design codes that *guarantee* that, in every rewrite, the cost is at most 1. That way, the code supports a large number of writes before block erasure. It was shown in [4] that codes with worst-case cost of 1 allows the writing of at most 1 bit per cell in each writing cycle.

A further generalization of the model was proposed in [5]. In this model, the cells need to induce a permutation of a given *multiset*. That is, each rank is occupied by a pre-determined number of cells, according to a specific multiset. For that model, it was shown in [5] that code with cost 1 can store up to 2 bits per cell in each cycle. Notice that this generalization doubles the amount of information storage for codes with cost 1. In addition, the generalization allows the rate to approach that of the non-binary write-once-memory model [8], when the number of writes and cell levels is high. In this work, we design rewriting codes with cost 1, that allow the writing of nearly 2 bits per cell in each cycle, and thus approach the limit of the model. Our construction takes advantage of the recently discovered polar codes, which were recently used in the construction of write-once-memory codes in [3].

The rest of the paper is organized as follows. In section II, we formally present the problem we study in this paper. In section III we give a background on polar WOM codes that serve in our construction. Section IV describes our construction of rank modulation codes. Finally, in section V, we give some concluding remarks.

II. Notations and Model

Consider a set of N cells, each taking one of q levels. Denote $\mathbf{c} = (c_1, c_2, \ldots, c_N)$, where $c_i \in \{0, 1, \ldots, q-1\}$, to be the *cell-state vector*. Denote a permutation of a multiset as a *multipermutation*, where the multiset is defined as following. Let m be the number of ranks, and let the number of cells in the i-th rank, $1 \leqslant i \leqslant m$, be denoted by z_i. z_i is also called the *multiplicity* of that rank. In the case that all multiplicities are equal, denote this number by z. Note that $N = \sum_{i=1}^{m} z_i$. Now let P_m be the set of all N-cells multipermutations $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(N))$ with m ranks. That is, for $1 \leqslant j \leqslant N$, $\sigma(j) \in \{1, \ldots, m\}$, and for $1 \leqslant i \leqslant m$, $\sigma^{-1}(i)$ is the set of all cells with rank i, i.e., $\sigma^{-1}(i) = \{j \mid \sigma(j) = i\}$. We call the vector $\mathbf{z} = \{z_1, z_2, \ldots, z_m\}$ a *multiplicity vector*. The set of all multipermutations of m ranks with multiplicity vector $\mathbf{z}$ is denoted by $P_{m,\mathbf{z}}$. Hence, $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(N)) \in P_{m,\mathbf{z}}$ if and only if for $1 \leqslant i \leqslant m$, $|\sigma^{-1}(i)| = z_i$. In case that $z = z_i$ for all $1 \leqslant i \leqslant m$, we denote the set $P_{m,\mathbf{z}}$ simply by $P_{m,z}$, and we follow the same analogy in the other definitions in the paper which include the multiplicity vector $\boldsymbol{z}$.

Given a cell-state vector $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ and a multiplicity vector $\mathbf{z} = \{z_1, z_2, \ldots, z_m\}$, the multipermutation $\sigma_{\mathbf{c},\mathbf{z}} = (\sigma(1), \sigma(2), \ldots, \sigma(N))$ is derived as follows. First, let $i_1, \ldots, i_N$ be an order of the cells such that $c_{i_1} \leqslant c_{i_2} \leqslant \cdots \leqslant c_{i_N}$. Then, the cells $i_1, \ldots, i_{z_1}$ get the rank 1, the cells $i_{z_1+1}, \ldots, i_{z_1+z_2}$ get the rank 2 and so on. More rigorously, for $1 \leqslant i \leqslant m$, the cells $i_{m_i}, i_{m_i}+1, \ldots, i_{M_i}$ get the rank i, where $m_i = 1 + \sum_{\ell=1}^{i-1} z_\ell$ and $M_i = \sum_{\ell=1}^{i} z_\ell$, i.e., $\sigma(i_{m_i}) = \sigma(i_{m_i}+1) = \cdots = \sigma(i_{M_i}) = i$. Note that a given cell-state vector can generate different multipermutations in case that there is equality between the levels of cells in adjacent ranks. In this case, we will define the multipermutation to be illegal and denote $\sigma_{\mathbf{c},\mathbf{z}} = F$. Given a multiplicity vector $\mathbf{z} = \{z_1, z_2, \ldots, z_m\}$, we let $Q_{\mathbf{z}}$ be the set of all cell-state vectors which result with a valid multipermutation, that is, $Q_{\mathbf{z}} = \{\mathbf{c} \in \{0, 1, \ldots, q-1\}^N \ |\sigma_{\mathbf{c},\mathbf{z}} \neq F\}$.

After a rewriting operation, the cell state is denoted as $\mathbf{c}' = (c'_1, c'_2, \ldots, c'_N)$. The cost of the rewriting operation is defined as $\max_i\{c'_i\} - \max_i\{c_i\}$, and the goal is to design a code that allows the writing of any information message with a rewrite cost of at most 1. We consider only the case where the encoder knows and the decoder does not know the previous state of the memory. The encoder and decoder use the same code for every cycle, and there are no decoding errors (zero-error case). For the cell states $\mathbf{c}$ and $\mathbf{c}'$, we denote $\mathbf{c} \leqslant \mathbf{c}'$ if and only if $c_i \leqslant c'_i$, for all $i = 1, 2, \ldots, N$. We are now ready to define the rewriting codes we study in this paper.

Definition 1. *An $(N, q, r, D, \mathbf{z} = (z_1, z_2, \ldots, z_m))$* ***rank-modulation rewriting code*** *is a coding scheme $\mathcal{C}(f, g)$ consisting of N q-level cells and a pair of encoding function f and decoding functions g. Let $I = \{1, \cdots, D\}$ be the set of input information symbols. The encoding function $f : I \times Q_{\mathbf{z}} \rightarrow Q_{\mathbf{z}}$, and the decoding function $g : Q_{\mathbf{z}} \rightarrow I$ satisfy the following constraints:*

1) *For any $d \in I$ and $\mathbf{c} \in Q_{\mathbf{z}}$, $\mathbf{c} \leqslant f(d, \mathbf{c})$.*
2) *For any $d \in I$ and $\mathbf{c} \in Q_{\mathbf{z}}$, $g(f(d, \mathbf{c})) = d$.*
3) *For any $\mathbf{c}_1, \mathbf{c}_2 \in Q_{\mathbf{z}}$, if $\sigma_{\mathbf{c}_1,\mathbf{z}} = \sigma_{\mathbf{c}_2,\mathbf{z}}$ then $g(\mathbf{c}_1) = g(\mathbf{c}_2)$.*
4) *For any $d \in I$ and $\mathbf{c} \in Q_{\mathbf{z}}$, $\mathbf{c}' \doteq f(d, \mathbf{c})$, $\max_i\{c'_i\} - \max_i\{c_i\} \leqslant r$.*

The rate of the code is $\mathcal{R} = (1/N)\log_2 D$.

It was shown in [5] that the maximal rate in this model is 2 bits/cell. In this work, we propose codes that approach this rate, with low complexity of encoding and decoding. In the next section we bring a short background on polar write once memory codes, that form an important ingredient in our code construction.

III. Polar WOM Codes

The method of channel polarization was first proposed by Arikan in his seminal paper [1], in the context of channel coding. We describe it here briefly by its application for coding on a write-once-memory, as proposed by Burshtein and Strugatski [3]. This application is based on the use of polar coding for lossy source coding, that was proposed by Korada and Urbanke [15].

Let $G_2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$, $G_2^{\otimes n}$ be its n-th Kronecker product, and $N = 2^n$. Consider a memoryless channel with a binary-input and transition probability $W(y|x)$. Define a vector $\boldsymbol{u} \in \{0,1\}^N$, and let $\boldsymbol{x} = \boldsymbol{u}G_2^{\otimes n}$, where the matrix multiplication is over $GF(2)$. The vector $\boldsymbol{x}$ is the input to the channel, and $\boldsymbol{y}$ is the output word. The main idea of polar coding is to define N sub-channels

$$W_N^{(i)}(\boldsymbol{y}, \boldsymbol{u}_1^{i-1}|u_i) = P(\boldsymbol{y}, \boldsymbol{u}_1^{i-1}|u_i) = \frac{1}{2^{N-1}} \sum_{\boldsymbol{u}_{i+1}^{N}} W(\boldsymbol{y}|\boldsymbol{u}),$$

where $\mathbf{u}_i^j$, for $1 \leqslant i < j \leqslant N$, denotes the subvector $(u_i, \ldots, u_j)$. For large N, each sub-channel is either very reliable or very noisy, and therefore it is said that the channel is polarized. A useful measure for the reliability of a sub-channel $W_N^{(i)}$ is its Bhattacharyya parameter, defined by

$$Z(W_N^{(i)}) = \sum_{y \in \mathcal{Y}} \sqrt{W_N^{(i)}(y|0) W_N^{(i)}(y|1)},$$

Consider now a memory consists of N binary valued cells, such that a cell of state ”0” can be changed into state ”1”, but a cell of state ”1” cannot be changed. This model is called Write Once Memory (WOM), since each cell can only be written once. The traditional WOM problem is how to write multiple times on the memory, and achieve high sum-rate. Nonetheless, we only present here the case of a single write to the memory, where the initial state already has cells with values of '1'. Assume that a user wishes to store information in the memory, where the encoder knowns the initial state of the memory, while the decoder doesn't. We further assume that there is no noise in the model. Let $\boldsymbol{s} \in \{0,1\}^N$ be the initial cell-state, and let p be the fraction of 1's in s. That is, $p = w(s)/N$, where $w(s)$ is the number of 1's in s. In addition, assume that a user wishes to store the message $\boldsymbol{a} \in \{0,1\}^k$. Note that in the case that the decoder *knows* the initial state $\boldsymbol{s}$, the communication rate of the memory is $\mathcal{R} = k/n = p$. Therefore, when the decoder doesn't know $\boldsymbol{s}$, the rate cannot exceed p. The following scheme allows a rate arbitrarily close to p for N sufficiently large.

Consider a binary erasure channel with erasure probability p. This channel is served as a *test channel*, in a compression scheme. Let X be a binary input to the channel, and (S, G) be the output, where S and G are binary variables as well. In the case of a successful use of the channel, $S = 1$, and $G = X$. In the case of erasure, $S = 0$, and G is uniformly distributed. The probability transition function of the channel can be written as

$$W((S,G)=(s,g)|X=x) = \begin{cases} p/2 & \text{if } s=0, \\ (1-p) & \text{if } s=1, g=x, \\ 0 & \text{if } s=1, g\neq x. \end{cases}$$

The channel is polarized by the sub-channels $W_N^{(i)}$, and a *frozen set* F is designed by

$$F = \left\{ i \in \{1,\ldots,N\} : Z(W_N^{(i)}) \geqslant 1 - 2\delta_N^2 \right\}, \qquad (1)$$

where $\delta_N = 2^{-N^{\beta}}/(2N)$, for any $0 < \beta < 1/2$. It was shown in [15] that $|F| = N(p-\delta)$, where δ is arbitrarily small for N sufficiently large.

Let $\hat{\boldsymbol{s}} = f_{\text{WOM}}(\boldsymbol{s}, \boldsymbol{a})$ be the WOM encoder. The encoder uses a common randomness source, also called *dither*, denoted by $\boldsymbol{g}$, sampled from an N dimensional uniformly distributed random binary vector, and known both to the encoder and to the decoder. Let $y_j = (s_j, g_j)$ and $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)$. The encoder creates a vector $\hat{\boldsymbol{u}} \in \{0,1\}^N$ in the following way. First, it sets $\boldsymbol{u}_F = \boldsymbol{a}$, where $\boldsymbol{u}_F$ is the vector of the elements of the vector $\boldsymbol{u}$ in the set F. Then, it compresses the vector $\boldsymbol{y}$ by the following successive cancellation scheme. For $i = 1, 2, \ldots, N$, let $\hat{u}_i = u_i$ if $i \in F$. Otherwise, let

$$\hat{u}_i = \begin{cases} 0 & \text{w.p } L_N^{(i)}/(L_N^{(i)}+1) \\ 1 & \text{w.p } 1/(L_N^{(i)}+1) \end{cases},$$

where

$$L_N^{(i)} = L_N^{(i)}(\boldsymbol{y}, \boldsymbol{u}_0^{i-1}) = \frac{W_N^{(i)}(\boldsymbol{y}, \boldsymbol{u}_0^{i-1}|u_i = 0)}{W_N^{(i)}(\boldsymbol{y}, \boldsymbol{u}_0^{i-1}|u_i = 1)}.$$

Finally, the encoder decompresses the resulting vector $\hat{\boldsymbol{u}}$ into $\boldsymbol{x} = \hat{\boldsymbol{u}} G_2^{\otimes n}$, and sets $\hat{\boldsymbol{s}} = \boldsymbol{x} + \boldsymbol{g}$ to be the new cell-state vector.

The decoder, $\boldsymbol{a} = g_{\text{WOM}}(\hat{\boldsymbol{s}})$, calculates $\boldsymbol{x} = \hat{\boldsymbol{s}} + \boldsymbol{g}$, and then recovers $\boldsymbol{a} = \left(\boldsymbol{x}(G_2^{\otimes n})^{-1}\right)_F$, where, again, $(\boldsymbol{b})_F$ denotes the elements of the vector $\boldsymbol{b}$ in the set F. Both the encoding and the decoding complexities are $O(N \log N)$. In [3], a few slight modifications for this scheme are described, for the sake of the proof. Note that the encoder is using a randomized algorithm and it might fail with a small probability. We present the following Lemma from [3], as it will serve us in the construction of rank-modulation codes.

Lemma 1. *[3] Consider the scheme described above. Then for any $\epsilon > 0, 0 < \beta < 1/2$ and N sufficiently large, the following holds w.p. $1 - 2^{-N^{\beta}}$,*

1) $|\{k : s_k = 0 \text{ and } \hat{s}_k = 1\}| < (p/2 + \epsilon)N$,
2) $\{k : s_k = 1 \text{ and } \hat{s}_k = 0\} = \emptyset$.

IV. Code Construction

For the simplicity of the presentation, assume that the cells are placed in consecutive levels, starting from $\ell_{\min}$. That is, for each rank $1 \leqslant i \leqslant m$ and cell $j \in \sigma^{-1}(i)$, $c_j = \ell_{\min} - 1 + i$. In addition, assume that for each rank i, $z_i = z$.

An important property of the construction is the fact the cost of most rewrites is 1. That is achieved by the following encoding function. First, increase the levels of the cells in rank 1 by 1. Notice that now $2z$ cell are in level $\ell_{\min} + 1$. Among these cells, choose z cells, according to some function of the input data, and increase their levels by 1. Now note that $2z$ cells are in level $\ell_{\min} + 2$. Again, choose z cells among them, and increase their levels by 1. Continue this way, until z cell are chosen out of the $2z$ cells in level $\ell_{\min} + m - 1$, and their levels are increased to $\ell_{\min} + m$, to finish the rewrite process. Notice that the level of the highest cells is now $\ell_{\min} + m$, while before the rewrite it was $\ell_{\min} + m - 1$, meaning that the cost of rewrite is 1. This is the framework of the encoding function. Notice that there are $m - 1$ selections, each time z cells are selected out of $2z$ candidate cells, according to some function of the input data. Our approach is to use a different *part* of the input data for each selection.

According to this framework, the value of $\mathbf{c}' = f(d, \mathbf{c})$ is encoded by a sequence of functions, each making a subset choice according to a *different* part of the input data d. Assume the input data d is partitioned into $m - 1$ parts and let $(d_1, d_2 \ldots, d_{m-1})$ be the data parts associated with each rank, where rank m doesn't represent any information. The first function determines the cells from $\sigma^{-1}_{\mathbf{c},\mathbf{z}}(1) \cup \sigma^{-1}_{\mathbf{c},\mathbf{z}}(2)$ which are assigned to be the set $\sigma^{-1}_{\mathbf{c}',\mathbf{z}}(1)$ as a function of the input data d_1. Thus we can write, $\sigma^{-1}_{\mathbf{c}',\mathbf{z}}(1) = f_1(d_1, \sigma^{-1}_{\mathbf{c},\mathbf{z}}(1) \cup \sigma^{-1}_{\mathbf{c},\mathbf{z}}(2))$, for some function f_1. Similarly, for $i = 2, 3, \ldots, m - 1$, there exists a function f_i such that

$$\sigma^{-1}_{\mathbf{c}',\mathbf{z}}(i) = f_i(d_i, \{\cup_{j=1}^{i+1} \sigma^{-1}_{\mathbf{c},\mathbf{z}}(j)\} \setminus \{\cup_{j=1}^{i-1} \sigma^{-1}_{\mathbf{c}',\mathbf{z}}(j)\}).$$

The decoder will operate in a similar way which will be explained in the sequel as part of the construction details.

For each $i = 1, \ldots, m - 1$

$$|\{\cup_{j=1}^{i+1} \sigma^{-1}_{\mathbf{c},\mathbf{z}}(j)\} \setminus \{\cup_{j=1}^{i-1} \sigma^{-1}_{\mathbf{c}',\mathbf{z}}(j)\}| = 2z.$$

Hence, in the encoding function f_i, if we consider the cells in the set $\{\cup_{j=1}^{i+1} \sigma^{-1}_{\mathbf{c},\mathbf{z}}(j)\} \setminus \{\cup_{j=1}^{i-1} \sigma^{-1}_{\mathbf{c}',\mathbf{z}}(j)\}$ as binary cells of value zero and all other cells of value one, then we can only program the zero cells to be one. Therefore, the key point in designing these encoding functions is to observe the similarity to the WOM problem that was described in section III. However, there is an important difference between the WOM problem and our problem of encoding a single rank. While in a WOM code there is no significance to the *number* of cells that are written, in our codes we seek to write such that *exactly* z cells will remain in level zero. Our approach to tackle that difference is to add extra redundancy cells in order to make the number of written cells exactly z w.h.p.. The number of redundancy cells is kept small, such that the rate can still be arbitrarily close to the upper bound.

While the number of redundancy cells can be made small, we still keep them as part of the cells in the multipermutation. That is, we still want to have a predefined number of cells in each rank. We do this in the following manner. In rank i, for each index of a flipped cell we want to store, we assign n' redundancy cells, where half of them are in rank i, and the other half in rank $i+1$.

Our construction uses an extension of Lemma 1. Note that according to Lemma 1, $w(\boldsymbol{s}') < (1-p+\epsilon)N$, where $w(\boldsymbol{s}')$ is the weight of $\boldsymbol{s}'$. It is possible to show, by the same proof used in this Lemma, that $w(\boldsymbol{s}') > (1-p-\epsilon)N$ also holds. Let us now describe the construction formally. To simplify the notation and representation of the construction we dropped all floors and ceilings, so some of the values are not necessarily integers as required. This may encounter a small lost in the rate of the code, but it will be minor and thus can be neglected.

Construction 1. *Let m, z, N be positive integers such that $N = mz$. Let $p = 2/m$ and $0 < \epsilon < p/2$. Let $N' = N + m\epsilon N n'$ (the value of n' will be explained later). The first N cells are called the information cells and are denoted by $\boldsymbol{c} = (c_1, \ldots, c_N)$. The last $r = m\epsilon N n'$ cells are called the redundancy cells and are partitioned into $m\epsilon N$ vectors $\boldsymbol{p}_{k,j}$ for $1 \leqslant k \leqslant m, 1 \leqslant j \leqslant \epsilon N$, each of n' cells. We assume that there is a function $h : \{1, 2, \ldots, N\} \to \{0,1\}^{n'}$ which receives an integer between 1 and N, and returns a balanced vector of length n'. h can be implemented, for example, by [14, pp. 5-6] or [13], where in both cases $\log N < n' < 2\log N$. We also assume that this function has an inverse function $h^{-1} : Im(h) \to \{1, 2, \ldots, N\}$.*

An $(N', q, 1, D, Z)$ rank-modulation rewriting code $\mathcal{C}$ is defined according to the following encoding function f_{RM} and decoding function g_{RM}. The number of messages on each write is $D = 2^{(2z-\delta N)(m-1)}$ and each message will be given as $m-1$ binary vectors, each of length $2z - \delta N$ bits. The cost of each rewrite is 1, and $Z = N'/m = z + \epsilon N n'$.

On the encoding and decoding functions, on each write we have the following assumptions:

1) *The information cells vector $\boldsymbol{c}$ and the redundancy cells vector $\boldsymbol{r}$ are multipermutations with m consecutive levels such that the number of cells in each level is the same. We let $\ell_{\min}$ be the minimum cell level and $\ell_{\max}$ be the maximum level (note that $\ell_{\max} - \ell_{\min} = m-1$).*
2) *We let $\sigma_{c,z}$ be the multipermutation derived from the information cells vector. For $1 \leqslant i \leqslant m$, let $S_i = \sigma_{\boldsymbol{c},z}^{-1}(i)$ (note that $|S_i| = z$).*
3) *There are $\epsilon N(m-1)$ auxiliary variables, called index variables and are denoted by $I_{k,j}$ for $1 \leqslant k \leqslant m-1, 1 \leqslant j \leqslant \epsilon N$. These index variables will be stored in the redundancy cells and they will indicate the information cells that their levels was intentionally changed during the encoding process.*

Encoding Function** $f_{\boldsymbol{RM}}(\boldsymbol{c}, \boldsymbol{p}, \boldsymbol{d}) = (\boldsymbol{c}', \boldsymbol{p}')$**:
Let $\boldsymbol{c}$ be the current information cells vector, $\boldsymbol{p} = (\boldsymbol{p}_{1,1}, \ldots, \boldsymbol{p}_{m,\epsilon N})$ be the current redundancy cells vector, and $\boldsymbol{d} = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{m-1})$ be the information vector, where each $\boldsymbol{d}_i$ is a vector of $(p-\delta)N = 2z - \delta N$ bits. The new updated information cells vector $\boldsymbol{c}' = (c_1', \ldots, c_N')$ is determined as follows. Let S_1' be the set $S_1' = S_1$.
***Encoding of the** k**-th rank,** $1 \leqslant k \leqslant m-1$:*

1) *Let $\boldsymbol{v}_k = (v_{k,1}, \ldots, v_{k,N}) \in \{0,1\}^N$ be the vector defined as follows: $v_{k,i} = 0$ if and only if $i \in S_k' \cup S_{k+1}$.*
2) *Let $\boldsymbol{u}_k = f_{WOM}(\boldsymbol{v}_k, \boldsymbol{d}_k)$. (Note that $\boldsymbol{u}_k$ satisfies*
   a) *$(1-p/2-\epsilon)N \leqslant w(\boldsymbol{u}_k) \leqslant (1-p/2+\epsilon)N$,*
   b) *$g_{p,\epsilon}(\boldsymbol{u}_k) = \boldsymbol{d}_k$,*
   c) *$\boldsymbol{v}_k \leqslant \boldsymbol{u}_k$.)*
3) *Let $w_k = w(\boldsymbol{u}_k) - (1-p/2)N$ ($|w_k| \leqslant \epsilon N$), and let $i_1, \ldots, i_{|w_k|}$ be the first $|w_k|$ indices in $S_k' \cup S_{k+1}$ whose value in $\boldsymbol{u}_k$ is equal to $(sign(w_k)+1)/2$. The vector $\boldsymbol{u}_k'$ is defined to be $u_{k,i_j}' = 1 - u_{k,i_j}$ for $1 \leqslant j \leqslant |w_k|$ and for all other indices i, $u_{k,i}' = u_{k,i}$ (note that $w(\boldsymbol{u}_k') = (1-p/2)N$). Set the indices $I_{k,j} = i_j$ for $1 \leqslant j \leqslant |w_k|$ and for $|w_k|+1 \leqslant j \leqslant \epsilon N$, $I_{k,j} = 0$.*
4) *Let $S_k^* = \{i | u_{k,i}' = 0\}$ and $S_{k+1}' = (S_k' \cup S_{k+1}) \setminus S_k^*$. For every $i \in S_k^*$, set $c_i' = \ell_{\min} + k$.*

Finally, for every $i \in S_m'$, set $c_i' = \ell_{\max} + 1$.

The new redundancy cells vector $\boldsymbol{p}' = (\boldsymbol{p}_{1,1}', \ldots, \boldsymbol{p}_{m,\epsilon N}')$ is determined as follows to store the $(m-1)\epsilon n$ indices. For $1 \leqslant k \leqslant m-1, 1 \leqslant j \leqslant \epsilon N$, let

$$\boldsymbol{p}_{k,j}' = (\ell_{\min} + k) \cdot \boldsymbol{1} + h(I_{k,j}).$$

Finally, for $1 \leqslant j \leqslant \epsilon n$, $\boldsymbol{p}_{m,j}' = \boldsymbol{p}_{m,j} + \boldsymbol{1}$.

***Decoding Function** $g_{\boldsymbol{RM}}(\boldsymbol{c}, \boldsymbol{p}) = \boldsymbol{d}'$**:** Let $\boldsymbol{c} = (c_1, \ldots, c_N)$ be the information cells vector and $\boldsymbol{p} = (\boldsymbol{p}_{1,1}, \ldots, \boldsymbol{p}_{m,\epsilon N})$ be the redundancy cells vectors. The information vector $\boldsymbol{d}' = (\boldsymbol{d}_1', \ldots, \boldsymbol{d}_{m-1}')$ is decoded as follows.*
First the indices $I_{k,j}$ for $1 \leqslant k \leqslant m-1$, $1 \leqslant j \leqslant \epsilon N$, are decoded to be

$$I_{k,j} = h^{-1}(\boldsymbol{p}_{k,j} - (\ell_{\min} + k - 1) \cdot \boldsymbol{1}).$$

***Decoding of the** k**-th rank,** $1 \leqslant k \leqslant m-1$:*

1) *Let $\hat{\boldsymbol{u}}_k' = (u_{k,1}, \ldots, u_{k,N}) \in \{0,1\}^N$ be the vector defined to be $\hat{u}_{k,i}' = 0$ if and only if $i \in S_k$.*
2) *The vector $\hat{\boldsymbol{u}}_k$ is defined as follows. For all $1 \leqslant j \leqslant \epsilon N$, if $I_{k,j} \neq 0$ then $\hat{u}_{k,I_{k,j}} = 1 - \hat{u}_{k,I_{k,j}}'$ and for all other indices i, $\hat{u}_{k,i} = \hat{u}_{k,i}'$.*
3) *$\boldsymbol{d}_k' = g_{WOM}(\hat{\boldsymbol{u}}_k)$.*

By the construction, we get that $r/N = \epsilon m n'$. To make this ratio arbitrarily small, we must let ϵ be a function of N. However, it is assumed in Lemma 1 that ϵ is constant. For that reason, we extend the Lemma for the case of non-constant ϵ.

Lemma 2. *When $\epsilon(N)$ is a function of N, the results of Lemma* 1 *hold for any $\epsilon > N^{\frac{\beta-1}{2}}$.*

The proof of Lemma 2 follows the same lines of the proof of Lemma 1, and is omitted for space limitations. This result allows us to prove the desired properties of Construction 1.

Theorem 1. *For any $0 < \beta < 1/2$ and m and z sufficiently large, the rank modulation rewriting code in Construction* 1 *can be used to write an arbitrary message of rate $\mathcal{R} < 2$ with cost* 1*, w.p. at least $1 - 2^{-N^{\beta}}$. The encoding and decoding complexities are $O(mN \log N)$.*

Proof:

By the construction, the cost of each rewrite is 1. We can express the rate in the following way:

$$\begin{aligned} \mathcal{R} &= (1/N') \log_2 D \\ &= \frac{(2z - \delta N)(m-1)}{N + m\epsilon N n'} \\ &= 2 \cdot \frac{m-1}{m} \cdot \frac{z - \delta z m/2}{z} \cdot \frac{1}{1 + \epsilon m n'} \end{aligned}$$

Setting $\epsilon = 1/N^{1/4}$ (the smallest possible by Lemma 2) and $\delta = 2/m^2$, and remembering that $n' < 2 \log N$, we get that

$$\begin{aligned} \mathcal{R} &> 2 \cdot (1 - 1/m)^2 \cdot \frac{1}{1 + 2\epsilon m \log(zm)} \\ &= 2 \cdot (1 - 1/m)^2 \cdot \frac{1}{1 + 2(m^3/z)^{1/4} \log(zm)} \end{aligned}$$

Therefore, $\mathcal{R}$ can take any value below 2 for large enough m and z, if $z/(m^3 \log^4(zm))$ is large enough as well. The probability of writing failure is achieved by the union bound. Each time f_{WOM} is applied, the probability of encoding failure is at most $2^{-N^{\beta}}$. f_{WOM} is applied $m-1$ times in each operation of the rank-modulation encoding, and therefore, for large enough N, the rank-modulation encoding is successful w.p. at least $1 - 2^{-N^{\beta}}$.

Finally, we prove the encoding and decoding complexities. According to [3], the complexities of f_{WOM} and g_{WOM} are both $O(N \log N)$. In each rank, we also apply h or h^{-1}, which can be performed in logarithmic time in N (see e.g. [14, pp. 5-6] and [13]). The functions h and h^{-1} are applied at most ϵN times on each rank, and thus don't affect the complexity. Finally, since f_{WOM} and g_{WOM} are applied for each rank, the encoding and decoding complexities are $O(mN \log N)$. ■

V. Conclusions

In this paper we present a rewriting coding scheme for rank modulation. The construction allows to write arbitrary message with cost 1, where the rate is asymptotically optimal. There are several open problems that can improve the understanding of the proposed scheme. First, it is of interest to determine the relation between the rate and the number of cells. In order to determine this, it is required to characterize the relation between the rate of polar codes and the number of sub-channels. In addition, the design of error correcting codes for this scheme is a broad open problem. A related attempt for the WOM model is proposed in [9].

VI. Acknowledgments

This work was partially supported by the NSF grants ECCS-0801795 and CCF-1217944, NSF CAREER Award CCF-0747415, BSF grant 2010075 and a grant from Intellectual Ventures.

References

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. on Inform. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[2] A. Barg and A. Mazumdar, "Codes in permutations and error correction for rank modulation," *IEEE Trans. on Inform. Theory*, vol. 56, no. 7, pp. 3158–3165, Jul. 2010.

[3] D. Burshtein and A. Strugatski, "Polar write once memory codes," in *Proceedings of the 2012 IEEE International Symposium on Information Theory, ISIT2012, Cambridge, MA, USA*, Jul. 2012, pp. 1982–1986.

[4] E. En Gad, A. Jiang, and J. Bruck, "Compressed encoding for rank modulation," in *Proceedings of the 2011 IEEE Int. Symp. on Inform. Theory, ISIT2011, St. Petersburg, Russia*, Aug. 2011, pp. 884–888.

[5] ——, "Trade-offs between instantaneous and total capacity in multi-cell flash memories," in *Proceedings of the 2012 IEEE Int. Symp. on Inform. Theory, ISIT2012, Cambridge, Massachusetts*, Jul. 2012, pp. 990–994.

[6] E. En Gad, M. Langberg, M. Schwartz, and J. Bruck, "Constant-weight gray codes for local rank modulation," *IEEE Trans. on Inform. Theory*, vol. 57, no. 11, pp. 7431–7442, Nov. 2011.

[7] F. Farnoud, V. Skachek, and O. Milenkovic, "Rank modulation for translocation correction," in *Proceedings of the IEEE International Symp. on Inform. Theory (ISIT)*, Jun. 2012, pp. 2988–2992.

[8] F.-W. Fu and A. J. Han Vinck, "On the capacity of generalized write-once memory with state transitions described by an arbitrary directed acyclic graph," *IEEE Trans. on Inform. Theory*, vol. 45, no. 1, pp. 308–313, Jan. 1999.

[9] A. Jiang, Y. Li, E. En Gad, M. Langberg, and J. Bruck, "Joint rewriting and error correction in write-once memories," in *Submitted to ISIT 2013*.

[10] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation for flash memories," *IEEE Trans. on Inform. Theory*, vol. 55, no. 6, pp. 2659–2673, Jun. 2009.

[11] A. Jiang, M. Schwartz, and J. Bruck, "Correcting charge-constrained errors in the rank-modulation scheme," *IEEE Trans. on Inform. Theory*, vol. 56, no. 5, pp. 2112–2120, May 2010.

[12] M. Kim, J. K. Park, and C. Twigg, "Rank modulation hardware for flash memories," in *Circuits and Systems (MWSCAS), 2012 IEEE 55th International Midwest Symposium on*, aug. 2012, pp. 294 –297.

[13] D. E. Knuth, "Efficient balanced codes," *IEEE Trans. on Inform. Theory*, vol. 32, no. 1, pp. 51–53, 1986.

[14] ——, *The Art of Computer Programming Volume 4, Fascicle 3*. Addison Wesley, 2005.

[15] S. B. Korada and R. Urbanke, "Polar codes are optimal for lossy source coding," *IEEE Trans. on Inform. Theory*, vol. 56, no. 4, pp. 1751–1768, Apr. 2010.

[16] I. Tamo and M. Schwartz, "Correcting limited-magnitude errors in the rank-modulation scheme," *IEEE Trans. on Inform. Theory*, vol. 56, no. 6, pp. 2551–2560, Jun. 2010.

[17] Z. Wang and J. Bruck, "Partial rank modulation for flash memories," in *Proceedings of the 2010 IEEE International Symposium on Information Theory (ISIT2010), Austin, TX, U.S.A.*, Jun. 2010, pp. 864–868.