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 [12]. 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 be 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 [12] 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 [14]. In addition, the scheme was also implemented in phase-change memories, where its data retention capability was verified experimentally [19].

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], [13], [21]. In addition, other variations of rank modulation were proposed and studied, such as [6], [22].

In this work we focus on the notion of *rewriting codes*, that were proposed for the rank-modulation scheme in [12], in order to *reuse* the memory between block erasures. It is desirable to minimize the usage of block erasures, since they are slow, power consuming and reduce the device reliability. 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 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 which *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], and studied also in [18]. 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 codes 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

978-1-4799-0446-4/13/$31.00 ©2013 IEEE

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. Section III describes our construction of rank modulation codes. In section IV we give a background on polar WOM codes and show how they can serve in our construction. 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 follows. A multiset $M = \{a_1^{z_1}, \ldots, a_m^{z_m}\}$ is a generalization of the underlying set $\{a_1, \ldots, a_m\}$, where each a_i appears z_i times. We call the elements of the underlying set *ranks*, and z_i is called the *multiplicity* of the rank a_i. The number of ranks is m, and the cardinality of the multiset is $N = \sum_{i=1}^{m} z_i$. Now let P_M be the set of all N-cell permutations $\sigma_M = (\sigma_M(1), \sigma_M(2), \ldots, \sigma_M(N))$ of the multiset M. That is, for $1 \leqslant j \leqslant N$, $\sigma_M(j) \in \{a_1, \ldots, a_m\}$, and for $i \in \{a_1, \ldots, a_m\}$, $\sigma_M^{-1}(i)$ is the set of all cells with rank i, i.e., $\sigma_M^{-1}(i) = \{j \mid \sigma(j) = i\}$. We also denote $S_i = \sigma_M^{-1}(i)$, where M is clear form the context, and note that $|S_i| = z_i$.

Given a cell-state vector $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ and a multiset $M = \{a_1^{z_1}, \ldots, a_m^{z_m}\}$, the permutation $\sigma_{\mathbf{c},M} = (\sigma_{\mathbf{c},M}(1), \sigma_{\mathbf{c},M}(2), \ldots, \sigma_{\mathbf{c},M}(N))$ 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 a_1, the cells $i_{z_1+1}, \ldots, i_{z_1+z_2}$ get the rank a_2 and so on. More rigorously, for $i \in \{a_1, \ldots, a_m\}$, the cells $i_{n_i}, i_{n_i}+1, \ldots, i_{N_i}$ get the rank i, where $n_i = 1 + \sum_{\ell=1}^{i-1} z_\ell$ and $N_i = \sum_{\ell=1}^{i} z_\ell$, 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.

After a rewriting operation, the cell state is denoted as $\mathbf{c}' = (c_1', c_2', \ldots, c_N')$, and we similarly define $\sigma_{\mathbf{c}',M}$ and S_i'. Denote ℓ_i as the highest level among the cells of rank i. That is,

$$\ell_i = \max_{j \in S_i} \{c_j\}.$$

In addition, define ℓ_i' in the same way for $\mathbf{c}'$. The cost of the rewriting operation is defined as $\ell_m' - \ell_m$. Given a current multipermutation $\sigma_{\mathbf{c},M}$ and a multipermutation to be written $\sigma_{\mathbf{c}',M}$, the cost of rewrite can be minimized by writing $\mathbf{c}'$ as follows. For $i = 2, 3, \ldots, m$, for every $j \in S_i'$, c_j' is set to be the maximum of c_j and $\ell_{i-1}' + 1$. When writing in this fashion, it was shown in [5] that the cost is at most $\max_{1 \leqslant i \leqslant N} [\sigma_{\mathbf{c},M}(i) - \sigma_{\mathbf{c}',M}(i)]$. The goal is to design a code that allows the writing of any information message with a rewrite cost of at most r. 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 $(N, r, D, M = \{1^{z_1}, \ldots, m^{z_m}\})$ **rank-modulation rewriting code** is a coding scheme $\mathcal{C}(f_{RM}, g_{RM})$ consisting of N cells, a multiset M and a pair of encoding function f_{RM} and decoding functions g_{RM}. Let $I = \{1, \cdots, D\}$ be the set of input information symbols. The encoding function $f_{RM} : I \times P_M \to P_M$, and the decoding function $g_{RM} : P_M \to I$ satisfy the following constraints:*

1) *For any $d \in I$ and $\sigma \in P_M$, $g_{RM}(f_{RM}(d, \sigma)) = d$.*
2) *For any $d \in I$ and $\sigma \in P_M$, let $\sigma' = f_{RM}(d, \sigma)$, $\max_{1 \leqslant i \leqslant N} [\sigma(i) - \sigma'(i)] \leqslant r$.*

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 $(r+1)\log_2(r+1) - r\log_2 r$ bits/cell, and specifically, when $r = 1$, the maximal rate is 2 bits/cell. In this work, we propose codes that approach this rate for any r, with low complexity of encoding and decoding. The construction is presented for the case of $r = 1$, where the generalization to any r is straight forward and will be discussed afterwards.

III. Code Construction

In this section we describe an $(N, 1, D, M = \{1^z, \ldots, m^z\})$ rank-modulation rewriting code with asymptotically-optimal rate. The first important idea of the code is that the encoding and decoding operations are divided into $m-1$ parts. The encoder perform the $m-1$ parts sequentially, while the decoder can perform them in parallel. In addition, the input message d is divided into $m-1$ parts, corresponding to the encoding and decoding parts, each contains $\frac{\log_2 D}{m-1}$ bits. Let d_i denote part i of the input message, for $i = 1, 2, \ldots, m-1$.

In each part $i = 1, 2, \ldots, m-1$ of the encoding function, the encoder determines the set of cell that will have rank i in the new multipermutation $\mathbf{c}'$. That is, in each part i, the encoder determines the set $S_i' = \sigma_M^{-1}(i)$. Note that since $M = \{1^z, \ldots, m^z\}$, $|S_i| = |S_i'| = z$ for all i. When the encoder is done with all the $m-1$ steps, the sets $S_1', S_2', \ldots, S_{m-1}'$ are all determined, and consequently, the set S_m' is also determined by the remaining cells.

In the first part, the encoder determines the set S_1' using a function $S_1' = f_r(S_1 \cup S_2, d_1)$. f_r selects S_1' as a subset of the set $S_1 \cup S_2$, according to the data part d_1. We denote the complement subset of S_1' as $S_2^* = (S_1 \cup S_2) \setminus S_1'$. The rest of the parts in the encoding function are performed similarly. In step i, we set $S_i' = f_r(S_i^* \cup S_{i+1}, d_i)$, and $S_{i+1}^* = (S_i^* \cup S_{i+1}) \setminus S_i'$. The decoding is done separately for each part, according to a function $d_i = g_r(S_i')$. To meet the first condition of a rank-modulation rewriting code, we must have $d_i = g_r(f_r(S, d_i))$ for any $S \in \{1, \ldots, N\}$ of size $2z$ and d_i. Given a scheme composed of such functions f_r and g_r, we need also to show that the cost of each rewrite is at most 1. To see this, notice that for each $j \in 1,2,\ldots,N$, $\sigma'(j) \leqslant \sigma(j)+1$, since S'_i is selected out of cells of rank at most $i+1$ in σ.

To design the functions f_r and g_r, we use a certain type of write-once-memory (WOM) codes, and add some adjustments to it. Write-once-memory is a memory with binary cells, where cells of state "1" cannot change their state to "0", but cells of state "0" can change their state to "1". WOM codes allow to store arbitrary information on the memory, when the encoder knows and the decoder doesn't know the initial state of the memory. The WOM codes are composed of an encoding function f_{WOM} and decoding function g_{WOM}, that serve as building blocks for f_r and g_r. We will show later two different implementations of $\{f_{\text{WOM}}, g_{\text{WOM}}\}$, but we first present an assumption on their behavior, and show how to construct a rank-modulation rewriting code using a scheme that meets this assumption.

Assumption 1. *For any $0 < p < 1$ and $0 < \epsilon < p/2$, there exists a binary code $\mathcal{C}_{WOM}$ of N cells with encoding function f_{WOM} and decoding function g_{WOM} such that given a cell-state vector $\boldsymbol{s} \in \{0,1\}^N$ with weight (number of 1's) $w(\boldsymbol{s}) = (1-p)N$, it is possible to write a binary vector $\boldsymbol{d}$ of $(p-\delta)N$ bits, for δ arbitrarily small and large enough N, such that the updated cell-state vector $\boldsymbol{s}' = f_{WOM}(\boldsymbol{s}, \boldsymbol{d})$ satisfies:*

1) $(1-p/2-\epsilon)N \leqslant w(\boldsymbol{s}') \leqslant (1-p/2+\epsilon)N$.
2) $\boldsymbol{s} \leqslant \boldsymbol{s}'$.
3) $g_{WOM}(\boldsymbol{s}') = \boldsymbol{d}$.

To connect the assumption to the scheme described above, let $p = 2/m = 2z/N$, and consider the vector $\boldsymbol{s}$ in the assumption to be the characteristic vector of the input sets in the functions f_r and g_r, such that $s_i = 0$ if cell i belongs to the input set, and $s_i = 1$ otherwise. Condition 2 of Assumption 1 implies that the set selected by f_{WOM} is a subset of the input set, as we need in f_r, since s'_i can be 0 (cell i is in S'_i) only if $s_i = 0$ (cell i is in the input set of f_{WOM}). Condition 3 is also exactly what we need for a decoding function g_f in the scheme above. Condition 1, however, does't meet exactly the requirement for f_r, since the output of f_r need to be of size exactly z, while condition 1 means that the size of the set selected by f_{WOM} is only *approximately* z. To overcome this issue, f_r "flips" a few bits in the output of f_{WOM}, to make its weight exactly $N-z$. The indices of the flipped cells are kept in additional redundancy cells, and since the number of those cells is small, the addition of the redundancy cells doesn't affect the asymptotic rate of the code. The decoder g_r first "flips back" the flipped bits, according to the redundancy cells, and then uses g_{WOM} to decode the stored message part.

One scheme that meets the conditions of Assumption 1 is the write-one-memory coding scheme that was recently proposed by Shpilka [20]. However, in this scheme the block-length N must be exponentially large in $1/\delta$, which results in high algorithmic complexity for rates close to 2 bits per cell. Alternatively, the polar WOM coding scheme of Burshtein and Strugatski [3] "almost" meets the conditions of Assumption 1, and can be implemented with N only polynomially large in $1/\delta$, as was shown recently in the context of channel polarization [9], [10]. We say that polar WOM codes "almost" meet the conditions of Assumption 1, since in polar WOM codes there is small probability that the encoding function fails to execute. However, since their block-length is shorter, they offer a more practical solution. The encoding failures can be solved by allowing a higher cost of rewriting in those rare occasions, or by the solutions proposed in [3]. We provide a background on polar WOM codes in section IV, and show how to use them to construct a scheme $\mathcal{C}_{\text{WOM}}$ as in Assumption 1.

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). We have the following notation:*

1) *The first N cells are called the information cells and are denoted by $\boldsymbol{c} = (c_1, \ldots, c_N)$. The information cells $\boldsymbol{c}$ represents a permutation $\sigma_c \doteq \sigma_{\boldsymbol{c},M_c}$ of the multiset $M_c = \{1^z, \ldots, m^z\}$.*
2) *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.*
3) *For $1 \leqslant k \leqslant m-1$, the cells $\boldsymbol{p}_{k,j}$ represents a permutation $\sigma_{k,j} \doteq \sigma_{\boldsymbol{p}_{k,j},M_k}$ of the multiset $M_k = \{k^{n'/2}, (k+1)^{n'/2}\}$.*
4) *The cells $\boldsymbol{p}_{m,j}$ represents a permutation $\sigma_{m,j} \doteq \sigma_{\boldsymbol{p}_{m,j},M_m}$ of the multiset $M_m = \{1^{n'/2}, m^{n'/2}\}$.*
5) *So in total the cells $\boldsymbol{p} = (\boldsymbol{p}_{1,1}, \ldots, \boldsymbol{p}_{m,\epsilon N})$ represents a permutation $\sigma_p \doteq \sigma_{\boldsymbol{p},M_p}$ of the multiset $M_p = \{1^{\epsilon N n'}, \ldots, m^{\epsilon N n'}\}$, and the entire N' cells represents a permutation $\sigma \doteq \sigma_{\{\boldsymbol{c},\boldsymbol{p}\},M}$ of the multiset $M = \{1^{z+\epsilon N n'}, \ldots, m^{z+\epsilon N n'}\}$.*

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' (a permutation of the multiset $\{0^{n'/2}, 1^{n'/2}\}$). h can be implemented, for example, by [16, pp. 5-6] or [15], 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 $(N', 1, D, Z)$ 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 $Z = N'/m = z + \epsilon N n'$. 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 ranks were intentionally changed during the encoding process.

Encoding Function $\sigma' = f_{\boldsymbol{RM}}(\sigma, \boldsymbol{d})$**:**
Let $\boldsymbol{d} = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{m-1})$ be the information vector, where each $\boldsymbol{d}_i$ is a vector of $(p-\delta)N = 2z - \delta N$ bits. σ' is composed of σ'_c, σ'_p and $\sigma'_{k,j}$ in the same manner as σ. The new updated information cells multipermutation σ_c is determined by the sets $S'_k = \sigma'^{-1}_c(k)$ for each rank k. Let $S_k = \sigma_c^{-1}(k)$, and 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 $\boldsymbol{u}_k = f_{WOM}(\boldsymbol{v}_k, \boldsymbol{d}_k)$.*
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 $S^*_{k+1} = (S_k^* \cup S_{k+1}) \setminus S'_k$.*

The new redundancy multipermutation σ'_{M_2} 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

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

Finally, for $1 \leqslant j \leqslant \epsilon n$, $\sigma'_{m,j} = \sigma_{m,j}$ (these cells are actually not needed except for clarifying the presentation).

Decoding Function $g_{\boldsymbol{RM}}(\sigma) = \boldsymbol{d}'$**:** *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

$$I_{k,j} = h^{-1}(\sigma_{k,j} - k \cdot \boldsymbol{1}).$$

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 $i \in S_k = \sigma_c^{-1}(k)$.*
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, in Assumption 1 we took ϵ to be constant. We show in Section IV that polar WOM codes meet Assumption 1 also in the case that ϵ is not constant.

We also note that the scheme can be modified easily for an (N, r, D, M) rank-modulation rewriting code, in order achieve a higher rate when allowing a higher cost of rewrite. To perform the modification, replace the set S_{k+1} in the steps 1,3 and 4 of the encoding of the k-th rank with the union of the sets $S_{k+1}, S_{k+2}, \ldots, S_{k+r}$. Finally, we present a Theorem that summarizes the properties of Construction 1. The Theorem assumes that $\mathcal{C}_{WOM}$ is implemented by polar WOM codes, and therefore it allows a small probability of failure. We omit the proof for space limitations.

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 $O(mN \log N)$.*

The next section describes the implementation of polar WOM codes.

IV. 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 for 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 [17].

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 write-once memory, as described in section III. 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 $\mathcal{R} = k/N = 1-p$. 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 $F = \left\{i \in \{1,\ldots,N\} : Z(W_N^{(i)}) \geqslant 1 - 2\delta_N^2\right\}$, where $\delta_N = 2^{-N^\beta}/(2N)$, for any $0 < \beta < 1/2$. It was shown in [17] that $|F| = N(1-p-\delta)$, 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 uses a randomized algorithm and it might fail with a small probability. We present the following Lemma from [3], that shows that the scheme described above meets the conditions of Assumption 1 with high probability.

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) $\{k : s_k = 1 \text{ and } \hat{s}_k = 0\} = \emptyset$.

As mentioned in section III, Construction 1 requires that Assumption 1 is met also when ϵ is not constant. For that reason, we extend Lemma 1 for this case.

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.

V. Conclusions

In this paper we presented a rewriting coding scheme for rank modulation. The construction allows to write arbitrary message with cost 1, where the rate is asymptotically optimal. An important open problem for this scheme is the design of error-correcting codes, with or without rewriting. A related attempt for the WOM model is proposed in [11].

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] V. Guruswami and P. Xia, "Polar codes: Speed of polarization and polynomial gap to capacity," Electronic Colloquium on Computational Complexity, Tech. Rep. 50, Apr. 2013. [Online]. Available: http://eccc.hpi-web.de/report/2013/050/

[10] S. H. Hassani, K. Alishahi, and R. Urbanke, "Finite-length scaling of polar codes," Apr. 2013. [Online]. Available: http://arxiv.org/abs/1304.4778

[11] 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*.

[12] 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.

[13] 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.

[14] 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.

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

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

[17] 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.

[18] Q. Li, "Compressed rank modulation," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, 2012, pp. 185–192.

[19] N. Papandreou, H. Pozidis, T. Mittelholzer, G. F. Close, M. Breitwisch, C. Lam, and E. Eleftheriou, "Drift-tolerant multilevel phase-change memory," in *Memory Workshop (IMW), 2011 3rd IEEE International*, 2011, pp. 1–4.

[20] A. Shpilka, "Capacity achieving multiwrite wom codes," in *Submitted to ISIT 2013*.

[21] 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.

[22] 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.