On the Capacity of Bounded Rank Modulation for Flash Memories

Zhiying Wang
Electrical Engineering Department
California Institute of Technology
Pasadena, CA 91125, USA
Email: zhiying@paradise.caltech.edu

Anxiao (Andrew) Jiang
Computer Science Department
Texas A&M University
College Station, TX 77843, USA
Email: ajiang@cs.tamu.edu

Jehoshua Bruck
Electrical Engineering Department
California Institute of Technology
Pasadena, CA 91125, USA
Email: bruck@caltech.edu

***Abstract*—Rank modulation has been recently introduced as a new information representation scheme for flash memories. Given the charge levels of a group of flash cells, sorting is used to induce a permutation, which in turn represents data. Motivated by the lower sorting complexity of smaller cell groups, we consider bounded rank modulation, where a sequence of permutations of given sizes are used to represent data. We study the capacity of bounded rank modulation under the condition that permutations can overlap for higher capacity.**

I. Introduction

Flash memory is an important non-volatile storage technology of wide applications. In flash memories, floating-gate cells use their charge-levels to store data [2]. For higher capacity, multi-level cells (MLCs) with an increasing number of levels are being developed. To increase a cell level, charge is injected into the cell by the Fowler-Nordheim tunneling mechanism or the hot-electron injection mechanism. This programming process is iterative to avoid over-injection. To lower any cell level, one must erase a whole cell block (typically 512K cells) and reprogram them starting at the lowest level. This asymmetric property caused by block erasure is a prominent feature of flash memories and presents a bottleneck of flash memories in terms of speed and reliability. There has been a number of recent works using the information theoretic approach to develop new storage schemes for flash memories. They include coding schemes for rewriting data [1] [4] [5] [6] [10], codes for correcting limited-magnitude errors [3], and the new rank modulation scheme for efficient and reliable cell programming and data storage [7] [8]. In this paper, we focus on and extend the rank modulation scheme.

Rank modulation is a new data representation scheme that uses the relative order of cell levels to represent data [7] [8]. Let $(c_1, c_2, \cdots, c_m)$ denote the charge levels of m cells, where each c_i (for $1 \leq i \leq m$) is an analog number and $\forall\ i \neq j$, $c_i \neq c_j$. Let $\mathcal{I}(c_1, c_2, \cdots, c_m) = (a_1, a_2, \cdots, a_m)$ be a function that induces from the charge levels a permutation, where for $i = 1, 2, \cdots, m$, $a_i = |\{j | a_j \leq a_i, j = 1, 2, \cdots, m\}|$. For example, if $m = 4$ and $(c_1, c_2, c_3, c_4) = (0.2, 0.3, 1.2, 0.5)$, then the induced permutation is $(a_1, a_2, a_3, a_4) = (1, 2, 4, 3)$. A group of m cells can store $\log_2(m!)$ bits of information. Since rank modulation uses permutations to represent data, the charge levels can take analog values instead of discrete values, making the programming process much more robust to over-injection and the stored data more robust to asymmetric errors.

In this paper, we study the capacity of rank modulation with bounded permutation sizes. To induce a permutation from a group of cells, a sorting algorithm of complexity $O(m \log m)$ is needed. Reducing the sorting complexity is important for the efficient hardware implementation of rank modulation. To study the capacity under this constraint, we propose a discrete model. Normalize the gap between the minimum and maximum charge levels of the memory to 1, and let δ denote the minimum charge difference to distinguish two levels. Then the largest possible size for a permutation is $D = \lfloor \frac{1}{\delta} \rfloor + 1$. However, in practice the permutation size should be smaller than D not only to reduce the sorting complexity, but also to make cell programming efficiently implementable. In this paper, we let $m \leq D$ denote the given permutation size (which is also the number of cells in a group), and study the achievable capacity. Each cell level is denoted by an integer in the set $\{1, 2, \cdots, D\}$. It should be noted that these D discrete numbers do not mean that in practice the charge levels are to be discrete instead of analog. They are used to derive the theoretical capacity under the considered constraints. When more constraints are introduced, the model can certainly be generalized.

An important observation is that by allowing cell groups to have overlaps (i.e., shared cells), the capacity can be improved. In this paper, we study this model, and explore the corresponding capacity. We present computational techniques and bounds for capacity, provide encoding and decoding techniques that achieve any rate smaller than the capacity, and compare the capacities of different schemes.

II. Bounded Rank Modulation

In this section, we define the basic concepts of bounded rank modulation. For convenience, for any two

integers a, b such that $a \leq b$, we define $[a, b] = \{a, a+1, \cdots, b\}$.

Let m and D be integers such that $m \leq D$. A *block* is a set of m cells whose levels are from the set $[1, D]$ and are all distinct. Let $(c_1, c_2, \cdots, c_m)$ denote those m cell levels. Then by definition, $c_i \in [1, D]$ for $i \in [1, m]$ and $\forall\ i \neq j$, $c_i \neq c_j$. For convenience, we call $(c_1, c_2, \cdots, c_m)$ a *block*, too, and call $\mathcal{I}(c_1, c_2, \cdots, c_m)$ the induced *permutation*. ($\mathcal{I}$ is as defined in the previous section.) If a block B induces a permutation P, then B is called a *realization* of P. Note that a permutation may have multiple realizations. For example, if $m = 6$ and $P = (1, 4, 3, 2)$, then both $(1, 6, 4, 3)$ and $(2, 5, 4, 3)$ are realizations of P.

Let $(c_1, c_2, \cdots, c_n)$ be the levels of n cells. Let $v < m$ be an integer and for convenience, let $(n - v)/(m - v)$ be an integer as well. For $i = 1, 2, \cdots, \frac{n-v}{m-v}$, let B_i denote the block $(c_{(i-1)(m-v)+1}, c_{(i-1)(m-v)+2}, \cdots, c_{(i-1)(m-v)+m})$. Note that the last v cell levels of B_i are also the first v cell levels of B_{i+1}, so we say these two blocks overlap by v. We say $(c_1, c_2, \cdots, c_n)$ is a *cell-level sequence* that consists of blocks that overlap by v, which we may also denote by $(B_1, B_2, \cdots, B_{(n-v)/(m-v)})$. For $i = 1, 2, \cdots, \frac{n-v}{m-v}$, let the m levels in B_i be all distinct. Then the sequence induces $(n - v)/(m - v)$ permutations $(P_1, P_2, \cdots, P_{(n-v)/(m-v)})$, where $P_i = \mathcal{I}(B_i)$ for $i = 1, 2, \cdots, \frac{n-v}{m-v}$. We call $(P_1, P_2, \cdots, P_{(n-v)/(m-v)})$ the induced *permutation sequence*, and call $(B_1, B_2, \cdots, B_{(n-v)/(m-v)})$ its *realization*. Again, a permutation sequence may have multiple realizations.

Definition 1 *(*BOUNDED RANK MODULATION $\mathcal{C}(n, m, D, v)$*) In a bounded rank modulation (BRM) code $\mathcal{C}(n, m, D, v)$, every codeword is a permutation sequence $(P_1, P_2, \cdots, P_{(n-v)/(m-v)})$ that has at least one realization. (The meaning of the parameters n, m, D, v is as presented above.) Let $|\mathcal{C}(n, m, D, v)|$ denote the number of codewords in code $\mathcal{C}$. Then, the* capacity *of the code is*

$$cap(\mathcal{C}) = \lim_{n\to\infty} \frac{\log |\mathcal{C}(n, m, D, v)|}{n}.$$

In general, allowing overlap between permutations can increase capacity. When there is no overlap (i.e., $v = 0$), the BRM code has capacity $\frac{\log m!}{m}$. When $v > 0$, the capacity may increase because every permutation consumes just $m - v$ cells on average.

III. BRM CODE WITH ONE OVERLAP AND CONSECUTIVE LEVELS

In this section, we study a special form of BRM code that allows efficient computation of its capacity. First, we present a computational method based on constrained systems.

Since $c_i \in [1, D]$ for $i \in [1, m]$, the BRM code is a *constrained system* over the alphabet S_m (the symmetric group on the set $[1, m]$). Define a *labeled graph* $G = (V, E, L)$ to be a directed graph with a state set V, an edge set $E \subseteq V \times V$ and an edge labeling $L : E \to S_m$. For $(u, v) \in E$, $L(u, v) = l$ is denoted by $u \xrightarrow{l} v$. G *represents* $\mathcal{C}$ if the set of all finite sequences obtained from reading the labels of paths in G equals the set of the codewords of $\mathcal{C}$. If the outgoing edges of each state are labeled distinctly, then G is *deterministic*. And G is *irreducible* if $\forall\ u, v \in V$, there is a path from u to v. Define $A_{|V|\times|V|}$ as the *adjacency matrix* of G, where A_{uv} equals the number of edges from u to v. In addition, suppose a deterministic graph G represents $\mathcal{C}(n, m, D, v)$ and $A_1, A_2, \cdots, A_k$ are the adjacency matrices of the irreducible components in G, then

$$cap(\mathcal{C}(n, m, D, v)) = \frac{\max_{1\leq i\leq k} \log \lambda(A_i)}{m - v} \quad (1)$$

where $\lambda(A)$ is largest positive eigenvalue of A [9].

Example 2 *A BRM code $\mathcal{C}(n, 2, 3, 1)$ can be represented by the graph G in Figure 1 (a). Each state represents the level of the current cell. $S_2 = \{12, 21\}$, the states are $V = \{1, 2, 3\}$, and the edges are $E = \{(i, i+1)|i = 1, 2\} \cup \{(i, i-1)|i = 2, 3\}$. The labeling is defined by $L(i, i+1) = 12, \forall\ i = 1, 2$ and $L(i, i-1) = 21, \forall\ i = 2, 3$. For example, the path along the states $1, 2, 3$, and 2 is a realization of the permutation sequence $(12, 12, 21)$. G is deterministic and irreducible. Hence, the adjacency matrix of G is*

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

By (1), the capacity is $\log(\lambda(A)) = 0.5$.

Notice in Example 2, the labeling L is essentially the ranks of the initial and terminal states of an edge. Also notice that every block $B_i = (c_i, c_{i+1})$ consists of two consecutive integers, i.e., $|c_i - c_{i+1}| = 1$. If we expand the idea of Example 2 to arbitrary $D \geq 2$ but keep $m = 2$, and $v = 1$, we will get the constrained system in Figure 1 (b). The adjacency matrix is

$$A = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \dots & \dots & 1 & 0 \end{pmatrix}_{D\times D}$$

The capacity is $\log \lambda(A) = \log(2\cos(\frac{\pi}{D+1}))$ [9].

We now formally define this type of constrained BRM code.

Definition 3 *(*BRM CODE WITH ONE OVERLAP AND CONSECUTIVE LEVELS $\mathcal{C}_I(n, m, D, 1)$*) For*

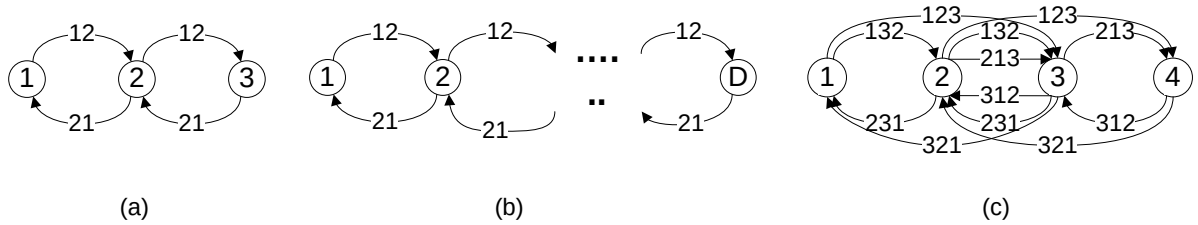

Fig. 1. Labeled graphs for $\mathcal{C}_I$. (a) $\mathcal{C}_I(n,2,3,1)$; (b) $\mathcal{C}_I(n,2,D,1)$ and D is arbitrary; (c) $\mathcal{C}_I(n,3,4,1)$.

the BRM code $\mathcal{C}_I(n,m,D,1)$, every codeword $(P_1,P_2,\cdots,P_{(n-1)/(m-1)})$ needs to satisfy the following additional constraint: the codeword has a realization $(B_1,B_2,\cdots,B_{(n-1)/(m-1)})$ such that for $i = 1,2,\cdots,\frac{n-1}{m-1}$, the m cell levels in the block B_i form a set of m consecutive numbers. That is, if $B_i = (c'_1,c'_2,\cdots,c'_m)$, then $\{c'_1,c'_2,\cdots,c'_m\} = [\min_{j=1}^m c'_j, \max_{j=1}^m c'_j]$.

In a labeled graph for $\mathcal{C}_I(n,m,D,1)$, each state corresponds to the charge level of an overlapped cell, so there are D states, $1,2,\cdots,D$. And each edge represents a permutation in a block $(c'_1,\cdots,c'_m)$. The first (or last) digit in an edge labeling corresponds to the initial (or terminal) state of the edge. Let $(a_1,\cdots,a_m) = \mathcal{I}(c'_1,\cdots,c'_m)$, then since each block has consecutive numbers, $\forall\ k,l \in [1,m]$,

$$c'_k - c'_l = a_k - a_l \tag{2}$$

For example, the labeled graph for $\mathcal{C}_I(n,3,4,1)$ is shown in Figure 1 (c).

The construction of the adjacency matrix for code $\mathcal{C}_I(n,m,D,1)$ is presented in the following theorem.

Theorem 4 *The adjacency matrix $A = (A_{ij})$ for $\mathcal{C}_I(n,m,D,1)$ has*

$$\begin{aligned} A_{ij} &= (m-2)!\min\{m-|i-j|, i, j, D-i+1, \\ &\quad D-j+1, D-m+1\} \end{aligned} \tag{3}$$

if $1 \le |i-j| \le m-1$, and $A_{ij} = 0$ otherwise.

Proof: A_{ij} indicates the number of permutations with $c'_1 = i, c'_m = j$. For fixed a_1 and a_m, there are $(m-2)!$ choices for $(a_2,\cdots,a_{m-1})$. Notice $i \to j$ only if $|a_1 - a_m| = |c'_1 - c'_m| \in [1,m-1]$. So $|\{(a_1,a_m)\}| \le m - |i-j|$, if $|i-j| \in [1,m-1]$. And $|\{(a_1,a_m)\}| = 0$ otherwise. If $i \in [1,m]$, then by (2), $\min_{1\le k\le m} c'_k = c'_1 - (a_1 - 1) = i - a_1 + 1 \ge 1$, which implies $a_1 \in [1,i]$, or $|\{a_1\}| = i$. Similarly, if $i \in [D-m+1, D]$, we will get $a_1 \in [m-D+i, m]$, or $|\{a_1\}| = D-i+1$. For $i \in [D-m+1, m]$, $a_1 \in [m-D+i, i]$, or $|\{a_1\}| = D-m+1$. And if $i \in [m, D-m+1]$, then $a_1 \in [1,m]$, or $|\{a_1\}| = m$. Hence, $|\{a_1\}| = \min\{i, D-i+1, D-m+1, m\}$. This argument also works for the terminal state j. Therefore, if $1 \le |i-j| \le m-1$,

$$\begin{aligned} A_{ij} &= (m-2)!|\{(a_1,a_m)\}| \\ &= (m-2)!\min\{m-|i-j|, |\{a_1\}|, |\{a_m\}|\} \\ &= (m-2)!\min\{m-|i-j|, i, j, D-i+1, \\ &\quad D-j+1, D-m+1\} \end{aligned}$$

And $A_{ij} = 0$ otherwise. ■

The capacity of $\mathcal{C}_I$ is $cap(\mathcal{C}_I) = \frac{\log\lambda(A)}{m-1}$. Some values of $cap(\mathcal{C}_I)$ and the capacity of the non-overlap code $\mathcal{C}(m,m,D,0)$ (for comparison) are shown in Figure 2.

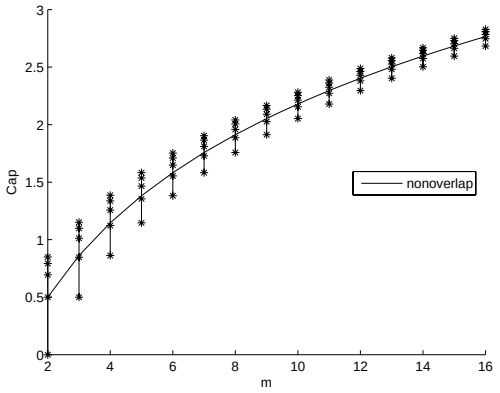

Fig. 2. Capacity for $\mathcal{C}_I$ (stars) and for the non-overlap code (solid line). The stars in each vertical line correspond to the same permutation size m, and $D = m, m+1, \cdots, m+4$ from bottom to top.

It is clear that the capacity of the code $\mathcal{C}_I(n,m,D,1)$ increases with D. And if $D \to \infty$, $cap(\mathcal{C}_I(n,m,D,1)) \to \frac{\log m!}{m-1}$, which is larger than the capacity of a non-overlapping code $\mathcal{C}(n,m,D,0)$. We now present a more general result.

Theorem 5 *For any $m \ge 2$ and $D \ge m+2$,*

$$cap(\mathcal{C}_I(n,m,D,1)) > cap(\mathcal{C}(n,m,D,0))$$

Proof: Notice $cap(\mathcal{C}(n,m,D,0)) = \log m!/m$, $\forall\ D \ge m$. If we proved $cap(\mathcal{C}_I(n,m,m+2,1)) > \log m!/m$, then this theorem is proved. When $m = 2,3$, $cap(\mathcal{C}_I(n,2,4,1)) = 0.6942 > \log 2!/2 = 0.5$ and $cap(\mathcal{C}_I(n,3,5,1)) = 1.0120 > \log 3!/3 = 0.8617$.

When $m \geq 4$, $D = m+2$, by (3), A is

$$(m-2)! \begin{pmatrix} 0 & 1 & 1 & 1 & \dots & 1 & 0 & 0 \\ 1 & 0 & 2 & 2 & \dots & 2 & 1 & 0 \\ 1 & 2 & 0 & 3 & \dots & 3 & 2 & 1 \\ 1 & 2 & 3 & 0 & \dots & 3 & 2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & 2 & 3 & 3 & \dots & 0 & 2 & 1 \\ 0 & 1 & 2 & 2 & \dots & 2 & 0 & 1 \\ 0 & 0 & 1 & 1 & \dots & 1 & 1 & 0 \end{pmatrix}_{(m+2)\times(m+2)}$$

By (1), it is necessary to find $\lambda(A)$. Let $B = \frac{1}{(m-2)!}A$, I be the identity matrix and x be an indeterminate variable. $det(B - xI) = 0$ implies $(-x-3)^{m-3}(x^2 + x - 1)f(x) = 0$, where $f(x) = -x^3 + (3m-8)x^2 + (7m-10)x + 3m - 3$. Thus $\lambda(B)$ is the largest positive root of $f(x)$. Notice $\forall\ x > \lambda(B)$, $f(x) < 0$, but $f(3m-6) = 3(m^2+m-5) > 0$, $m \geq 4$. So $\lambda(B) > 3m-6$, and $\lambda(A) > (3m-6)(m-2)!$. Now we are left to show

$$\frac{\log \lambda(A)}{m-1} > \frac{\log(3(m-2)(m-2)!)}{m-1} \geq \frac{\log m!}{m}$$

which is equivalent to $\frac{3^m(m-2)^m(m-2)!}{m^{m-1}(m-1)^{m-1}} \geq 1$. Notice $m \geq 4$, $\left(1 - \frac{1}{m}\right)^m \geq \frac{1}{e}$, and Stirling's Approximation, $m! \geq \sqrt{2\pi m}(m/e)^m$, thus

$$\begin{aligned} & \frac{3^m(m-2)^m(m-2)!}{m^{m-1}(m-1)^{m-1}} \\ = & \frac{3^m(m-1)(m-2)!}{m^{m-1}} \left(\frac{m-2}{m-1}\right)^{m-1} \frac{m-2}{m-1} \\ \geq & \frac{3^m(m-1)!}{m^{m-1}} \cdot \frac{1}{e} \cdot \frac{1}{2} \\ \geq & \frac{1}{2e} \frac{3^m\sqrt{2\pi(m-1)}}{e^{m-1}} \frac{(m-1)^{m-1}}{m^{m-1}} \\ \geq & \frac{1}{2e} \left(\frac{3}{e}\right)^m \sqrt{2\pi(m-1)} \geq 1 \end{aligned}$$

Thus the proof is completed. ∎

IV. BRM Code with One Overlap

We now consider the general BRM code with one overlap, $\mathcal{C}(n,m,D,1)$, which does not have the additional constraint of code $\mathcal{C}_I(n,m,D,1)$.

In this case, the cell levels of a block, $\{c'_1, \cdots, c'_m\}$, can be any set Q such that $Q \subseteq \{1,2,\cdots,D\}$ and $|Q| = m$. The labeled graph H generated is not deterministic in general. However, we are able to find a deterministic graph G that is equivalent to H (Lemma 2.1 in [9]). Here is an example.

Example 6 *The labeled graph H of $\mathcal{C}(n,2,4,1)$ is shown in Figure 3 (a). This is not deterministic since state 1 has 3 outgoing edges labeled 12. Let G be the deterministic representation of $\mathcal{C}$, then the states $V(G)$ are subsets of $V(H)$. And for $u, v \in V(G)$, $u \xrightarrow{l} v$ if $\forall\ j \in v, \exists\ i \in u$ and $i \xrightarrow{l} j$. So the resulting graph G is as shown in Figure 3 (b). States $\{2\}$, $\{3\}$, $\{1,3\}$, etc., have only outgoing edges, so their capacities are 0. Therefore the irreducible component of G maximizing $\lambda(A_i)$ is as in Figure 3 (c). By (1)we can then get $cap(\mathcal{C}(n,2,4,1)) = \log \lambda(A_i) = 0.8791 > cap(\mathcal{C}_I(n,2,4,1)) = 0.6942$.*

In general, suppose the deterministic graph G represents $\mathcal{C}(n,2,D,1)$, and A_i is the adjacency matrix for the irreducible component of G that has the largest eigenvalue. Then $\lambda(A_i)$ is the largest positive root of $-x^D + 2x^{D-1} - 1 = 0$. Comparing $cap(\mathcal{C}_I)$ and $cap(\mathcal{C})$, we have Figure 4. It can be seen that $cap(\mathcal{C})$ tends to 1 faster than $cap(\mathcal{C}_I)$, since it makes better use of the levels provided.

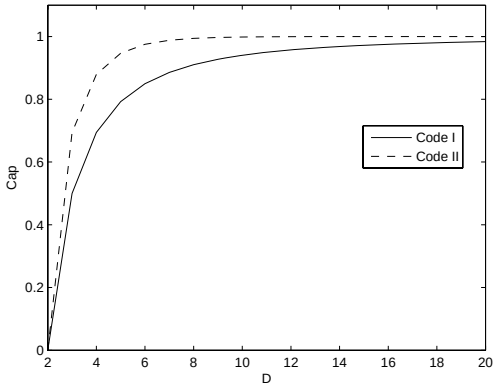

Fig. 4. Capacity for $\mathcal{C}$. The solid and dashed lines show capacity for $\mathcal{C}_I$ and $\mathcal{C}$, respectively.

The construction in the above example can be naturally extended to the case $m > 2$.

Encoders and decoders for BRM codes can be constructed as in figure 5. In the encoding process, the input information sequence $(x_0, x_1, \dots)$ is first encoded into a permutation sequence, $(P_0, P_1, \dots)$, satisfying the maximum cell level constraint, which is further mapped to a cell-level sequence, $(c_1, c_2, \dots)$. At last, the cell-level sequence is programmed into flash memory. And the decoder reverses this process by reading the cell levels, forming a permutation sequence, and at last retreiving the information.

- Permutaion encoder/decoder

Let $\mathcal{C}$ be a constrained system with a representation G, and p,q be positive integers. The q-th power of $\mathcal{C}$, $\mathcal{C}^q$, is represented by the labeled graph G^q, with the same set of states as G and edges/labelings corresponding to paths/labelings of legnth q in G. A *finite-state encoder* with rate $p:q$ is a lossless labeled graph H such that $H \subseteq G^q$ and each state of H has out-degree 2^p. We can then assign 2^p input tags (or p binary information bits) to the outgoing edges of each state. An encoder is (m,a)-*sliding-block decodable* if the i-th input tag in

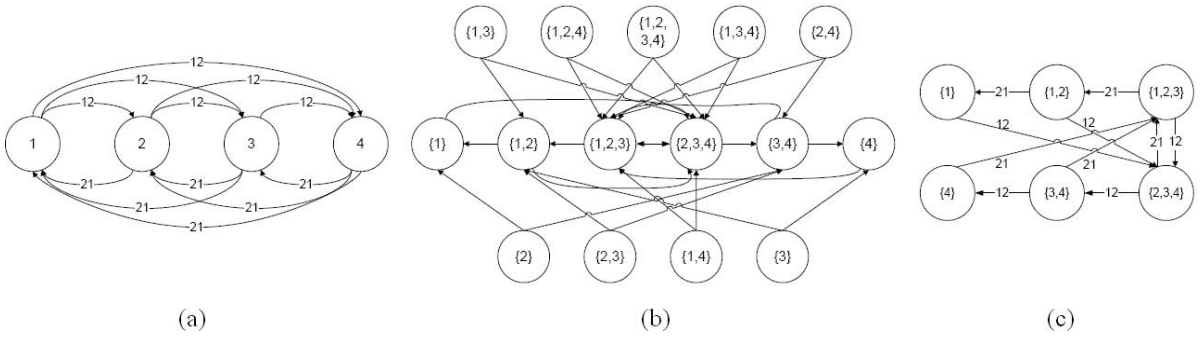

Fig. 3. Labeled graphs for $\mathcal{C}(n,2,4,1)$. (a) Labeled graph; (b) deterministic graph; (c) irreducible graph.

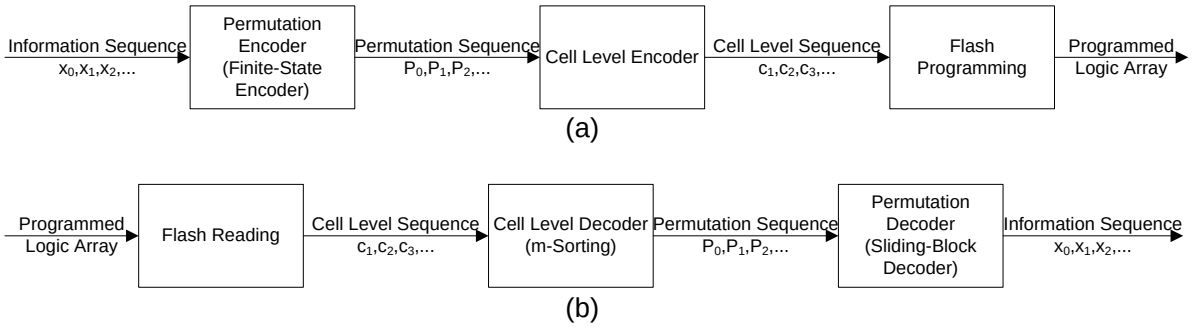

Fig. 5. (a)Encoder and (b) decoder for BRM codes

an input tag sequence is uniquely determined by the q-block labeling sequence $P_{i-m}^q, P_{i-m+1}^q, \ldots, P_i^q, \ldots, P_{i+a}^q$. The following theorem states that the capacity of any constrained system is always achievable [9].

Theorem 7 *Let $\mathcal{C}(n,m,D,v)$ be a constrained system and $p/q < (m-v)cap(\mathcal{C}(n,m,D,v))$ for positive integers p and q. Then there exists a sliding-block decodable finite-state encoder with rate $p:q$ and permutation encoding rate $p/(q(m-v))$.*

Theorem 7 can be proved by explicit constructions of encoders, such as the state-splitting algorithm [9]. And a sliding-block decoder is essentially a mapping from $(m+a+1)$ q-block labelings to a p-block binary input tag. Notice in the decoding process, decoding delay and error propagation is controlled within $m+a+1$ q-blocks, which depends on the construction of the encoder/decoder.

Example 8 *Continuing Example 6, take $p=3$ and $q=4$, then $p/q < cap(\mathcal{C}(n,2,4,1)) = 0.8792$. The 4-th power of $\mathcal{C}(n,2,4,1)$ has adjacency matrix*

$$A_i^4 = \begin{pmatrix} 4 & 2 & 1 & 1 & 2 & 3 \\ 3 & 2 & 1 & 1 & 1 & 3 \\ 2 & 1 & 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 & 1 & 2 \\ 3 & 1 & 1 & 1 & 2 & 3 \\ 3 & 2 & 1 & 1 & 2 & 4 \end{pmatrix}$$

After deleting States 3 and 4, and some edges from the graph, we get a finite-state encoder in Figure IV (a), which has out-degree $2^p = 8$ for each state, and each labeling block has size $q=4$. The notation $u \overset{x^3/P^4}{\rightarrow} v$ means the 4-block labeling P^4 is assigned the binary input tag x^3. For convenience, we denote the permutation $(1,2)$ by 1, and $(2,1)$ by 0 in the labeling. After merging the States 1 and 6, it is further simplfied as in Figure IV (b).

Let State 1 be the initial state for the encoding. Divide the input binary information bits into blocks of 3, and for any $x_i^3 = (x_{3i}, x_{3i+1}, x_{3i+2})$, encode it as the corresponding 4-block labeling, $P_i^4 = (P_{4i}, P_{4i+1}, P_{4i+2}, P_{4i+3})$.

Notice that in the encoding process, each 4-block labeling corresponds to only one input tag, independent of starting state. Therefore, we can construct a $(0,0)$-sliding-block decoder: for each received permutation block P_i^4 decode it to the unique information block $x_i^3 = (x_{3i}, x_{3i+1}, x_{3i+2})$, which equals to $(P_{4i+1}, P_{4i+2}, P_{4i+3})$ if $P_{4i} \neq P_{4i+1}$ and equals to $(P_{4i+3}, P_{4i+3}, P_{4i+3})$, otherwise.

- Cell-level encoder/decoder

Before programming into flash memories, we must first encode the permutation sequence $(P_0, P_2, \ldots)$ into a cell-level sequence $(c_1, c_2, \ldots)$, such that it induces this permutation sequence and does not exceed the maximum level. The following construction provides such an encoding. Let $P_i = (a_1^i, a_2^i, \ldots, a_m^i)$ and denote $c_{i(m-1)+j}$ by c_j^i, for $i \geq 0$, $j = 1,2,\ldots,m$. Then for BRM codes

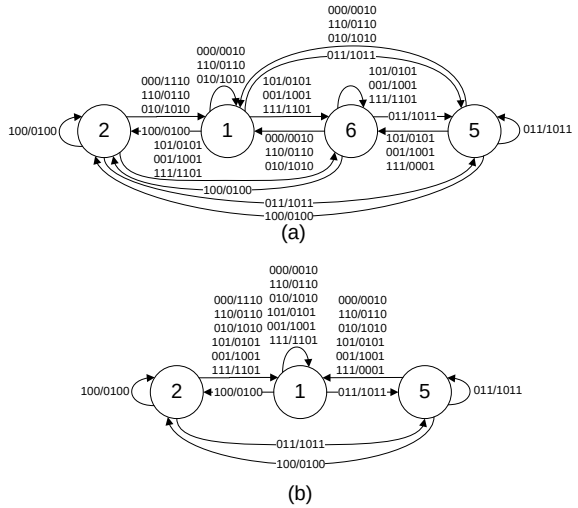

Fig. 6. Rate 3 : 4 finite-state encoder for $\mathcal{C}(n,2,4,1)$. (a) Labeled subgraph with out-degree 8, and (b) simplified encoder.

with one overlap, c_j^i is the cell level of a_j^i, for $i \geq 0$, $j = 1,2,\ldots,m$, and $c_1^i = c_m^{i-1}$, for $i \geq 1$.

Construction 9 *Let $(P_0,\ldots,P_i,\ldots)$ be the input permutation sequence of the cell-level encoder. Then the following assignment of $c_1^0, c_2^0, \ldots, c_{m-1}^0, \ldots, c_1^i, c_2^i, \ldots, c_{m-1}^i, \ldots$ defines a cell-level encoder.*

$$c_1^0 = \begin{cases} a_1^0, & \text{if } a_1^0 < a_m^0 \\ D - m + a_1^0, & \text{if } a_1^0 > a_m^0 \end{cases}$$

For $i \geq 1$,

$$c_1^i = \begin{cases} \max(a_1^i, c_1^{i-1} + a_m^{i-1} - a_1^{i-1}), \\ \quad \text{if } a_1^{i-1} < a_m^{i-1}, a_1^i < a_m^i \\ \max(a_1^i, a_m^{i-1}), \\ \quad \text{if } a_1^{i-1} > a_m^{i-1}, a_1^i < a_m^i \\ \min(D - m + a_1^i, D - m + a_m^{i-1}), \\ \quad \text{if } a_1^{i-1} < a_m^{i-1}, a_1^i > a_m^i \\ \min(D - m + a_1^i, c_1^{i-1} + a_m^{i-1} - a_1^{i-1}), \\ \quad \text{if } a_1^{i-1} > a_m^{i-1}, a_1^i > a_m^i \end{cases} \tag{4}$$

For $i \geq 0$ and $j = 2,\ldots,m-1$,

$$c_j^i = \begin{cases} c_1^i + a_j^i - a_1^i, \\ \quad \text{if } a_1^i < a_m^i, a_j^i < a_m^i \text{ or } a_1^i > a_m^i, a_j^i > a_1^i \\ c_1^{i+1} + a_j^i - a_m^i, \\ \quad \text{if } a_1^i < a_m^i, a_j^i > a_m^i \text{ or } a_1^i > a_m^i, a_j^i < a_1^i \end{cases} \tag{5}$$

When $m = 2$, using 1 and 0 to represent the permutations $(1,2)$ and $(2,1)$, respectively, the above construction is reduced to

$$c_1 = \begin{cases} 1, & \text{if } P_0 = 1 \\ D, & \text{if } P_0 = 0 \end{cases}$$

and for $i \geq 1$,

$$c_{i+1} = \begin{cases} c_i + 1 & \text{if } P_{i-1} = 1, P_i = 1 \\ 1 & \text{if } P_{i-1} = 0, P_i = 1 \\ D & \text{if } P_{i-1} = 1, P_i = 0 \\ c_i - 1 & \text{if } P_{i-1} = 0, P_i = 0 \end{cases}$$

One can check that if a permutation sequence satisfies the constraints for $\mathcal{C}(n,2,D,1)$, ie., there are at most $D-1$ zeros (or ones) between any two successive ones (or zeros), then the generated cell-level sequence realizes this sequence and has cell levels in $[1,D]$.

We now show that Construction 9 generates a cell-level realization for each codeword in BRM code.

Theorem 10 *Let $(P_0,P_1,\ldots,P_n)$ be a codeword in $\mathcal{C}((m-1)n+1,m,D,1)$ and $C = (c_1^0, c_2^0, \ldots, c_{m-1}^0, \ldots, c_1^{n-1}, c_2^{n-1}, \ldots, c_{m-1}^{n-1}, c_1^n)$ be the cell-level sequence generated in Construction 9. Then C is a realization of $P = (P_0,P_1,\ldots,P_{n-1})$. In particular, each cell level ranges between 1 and D.*

Proof: From (5) it is clear that $(c_1^i, c_2^i, \ldots, c_{m-1}^i, c_1^{i+1})$ has ranks $P_i = (a_1^i, a_2^i, \ldots, a_m^i)$. We are left to show that C ranges between 1 and D.

Assume $(d_1^0, d_2^0, \ldots, d_{m-1}^0, \ldots, d_1^n, d_2^n, \ldots, d_{m-1}^n, d_1^{n+1})$ is an arbitrary realization of P and $1 \leq d_j^i \leq D$ for $j = 1,\ldots,m-1$, $i = 0,1,\ldots,n$. For c_1^i, $i = 0,1,\ldots,n-1$, we will prove a stronger condition:

$$\begin{aligned} &a_1^i \leq c_1^i \leq d_1^i, a_m^{i-1} \leq c_1^i, \text{ if } a_1^i < a_m^i \\ &d_1^i \leq c_1^i \leq a_1^i + D - m, c_1^1 \leq a_m^{i-1} + D - m, \text{ if } a_1^i > a_m^i \end{aligned} \tag{6}$$

The inequalities containing a_m^{i-1} are not considered when $i = 0$. Notice the cell-level sequence $(d_1^i,\ldots,d_{m-1}^i,d_1^{i+1})$ induces $(a_1^i,\ldots,a_{m-1}^i,a_m^i)$. Hence, for $1 \leq j,k \leq m$ such that $a_j^i \geq a_k^i$, we have

$$a_j^i - a_k^i \leq d_j^i - d_k^i \tag{7}$$

Let $a_j^i = a_1^i, a_k^i = 1$, and $a_j^i = m, a_k^i = a_1^i$, and we get

$$a_1^i \leq d_1^i \leq a_1^i + D - m \tag{8}$$

Similarly, since $(d_1^{i-1},\ldots,d_{m-1}^{i-1},d_1^i)$ induces $(a_1^{i-1},\ldots,a_{m-1}^{i-1},a_m^{i-1})$, we have $a_m^{i-1} \leq d_1^i \leq a_m^{i-1} + D - m$. Suppose (6) holds, then $1 \leq a_1^i \leq c_1^i \leq a_1^i + D - m \leq D$ and $1 \leq a_m^{i-1} \leq c_1^i \leq a_m^{i-1} + D - m \leq D$. And by (5), either $1 \leq a_j^i = a_1^i + a_j^i - a_1^i \leq c_j^i = c_1^i + a_j^i - a_1^i \leq (a_1^i + D - m) + a_j^i - a_1^i = a_j^i + D - m \leq D$ or $1 \leq a_j^i \leq c_j^i = c_1^{i+1} + a_j^i - a_m^i \leq a_j^i + D - m \leq D$, which would complete the proof.

we will prove (6) by induction. For the base case($i = 0$), if $a_1^0 < a_m^0$, then $c_1^0 = a_1^0 \leq d_1^0$. And if $a_1^0 > a_m^0$, then $d_1^0 \leq a_1^0 + D - m = c_1^0$. Now suppose (6) holds for $i - 1 \geq 0$. If $a_1^{i-1} < a_m^{i-1}$ and $a_1^i < a_m^i$, then by induction,

$$a_1^{i-1} \leq c_1^{i-1} \leq d_1^{i-1}, \quad a_m^{i-2} \leq c_1^{i-1} \tag{9}$$

Now either $a_m^{i-1} \overset{(9)}{\leq} c_1^{i-1} + a_m^{i-1} - a_1^{i-1} \leq a_1^i = c_1^i \overset{(8)}{\leq} d_1^i$ or $a_1^i \leq c_1^{i-1} + a_m^{i-1} - a_1^{i-1} = c_1^i \overset{(9)}{\leq} d_1^{i-1} + a_m^{i-1} - a_1^{i-1} \overset{(7)}{\leq} d_1^i$ and $a_m^{i-1} \overset{(9)}{\leq} c_1^{i-1} - a_1^{i-1} + a_m^{i-1} = c_1^i$. Therefore, (6) holds for c_1^i in this case. If $a_1^{i-1} > a_m^{i-1}$ and $a_1^i < a_m^i$, then by (8) either $a_m^{i-1} \leq a_1^i = c_1^i \leq d_1^i$ or $a_1^i \leq a_m^{i-1} = c_1^i \leq d_1^i$. Therefore, (6) holds in this case, too. And we can prove by similar arguments that (6) is true for the other two cases, thus complete this proof. ■

The cell-level decoder is an m-sorter that orders every m-cell-level tuple associated with the permutations. The complexity is $m \log m$.

- Flash programming/reading

To avoid over-programming, the programming of a cell-level sequence into flash memories is operated from the lowest to the highest rank. Such a programming method will lead to a delay of n in the worst case. However, since for BRM code with one overlap, only c_1^i, $i \geq 0$ are contained in two permutations, we only need to obtain correct ordering for $\{c_1^i, c_1^{i+1}\}$ and $\{c_1^i, c_2^i, \ldots, c_{m-1}^i, c_1^{i+1}\}$, for all $i \geq 0$. And ordering of other sets of cell levels are not used in the code. Construction 11 follows the above idea and has smaller delay than n, for n sufficiently long.

Construction 11 *we first define a writing operation for c_1^i, denoted by $Op(i)$, as follows.*

1) *If $i \geq 1$, compare c_1^i and c_1^{i-1}. If $c_1^i < c_1^{i-1}$, write into flash $\{c_j^{i-1} | j = 2, \ldots, m-1, c_j^{i-1} < c_1^i\}$ from low to high level. Otherwise, write $\{c_j^{i-1} | j = 2, \ldots, m-1, c_1^{i-1} < c_j^{i-1} < c_1^i\}$ from low to high level.*
2) *If $c_1^i < c_1^{i+1}$, write into flash $\{c_j^i | j = 2, \ldots, m-1, c_j^i < c_1^i\}$ from low to high level. If $c_1^i > c_1^{i+1}$, write $\{c_j^i | j = 2, \ldots, m-1, c_1^{i+1} < c_j^i < c_1^i\}$ from low to high level.*
3) *Write c_1^i so that it has a higher charge level than all the cells written in steps 1 and 2.*
4) *If $i \geq 1$, and $c_1^i > c_1^{i-1}$, then write $\{c_j^{i-1} | j = 2, \ldots, m-1, c_j^{i-1} > c_1^i\}$ from low to high level. And if $c_1^i > c_1^{i+1}$, write into flash $\{c_j^i | j = 2, \ldots, m-1, c_j^i > c_1^i\}$ from low to high level.*

In steps 1, 2, and 4, the only requirement is that the charge level of each programmed cell is higher than the previously written one.

Now starting from $i = 0$, if $c_1^i < c_1^{i+1}$,do $Op(i)$, and update i with $i+1$. Otherwise, find the smallest $e \geq 1$, such that $c_1^{i+e} < c_1^{i+e+1}$, do $Op(i+e), Op(i+e-1), \ldots, Op(i)$, and then update i with $i+e+1$.

The above procedure clearly realizes the given permutation sequence, and moreover has a worst-case delay $D(m-1)$. In the worst case, $c_1^i = D, c_1^{i+1} = D-1, \ldots, c_1^{i+D-1} = 1$, and $c_1^{i-1} < c_2^{i-1} < D$, for some $i \geq 1$, hence one has to receive $c_2^{i-1}, \ldots, c_{m-1}^{i-1}, c_1^i, \ldots, c_{m-1}^i, \ldots, c_1^{i+D-1}$ to write c_2^i.

For BRM code with permutations of size 2, the programming process reduces to operations on c_1^i only, for $i \geq 0$, with maximum delay D.

Flash reading can be simply realized by sequentially reading off cell charge levels from flash.

Notice that both the cell-level encoder and flash programming has rate 1, so the BRM code encoder has rate $p/(q(m-v))$ by Theorem 7, which can be arbitrarily close to the capacity.

We will now give an example of the complete encoding/decoding process for BRM code $\mathcal{C}(n, 2, 4, 1)$. The encoder has rate 0.75 in this example.

Example 12 *Encoding: suppose the information sequence is* $(x_0 \ldots x_{11} \ldots) = (001\ 100\ 000\ 010 \ldots)$. *Then following Example 8, the encoded permutation sequence is* $(P_0 \ldots P_{15}) = (1001\ 0100\ 1110\ 1010 \ldots)$, *and the state transition in the finite-state encoder is State* $1 \to 1 \to 2 \to 1 \to 1$. *Using Construction 9, we get the encoded cell-level sequence* $(c_1 \ldots c_{16} \ldots) = (1\ 431\ 41\ 431\ 2\ 3\ 41\ 41\ 4 \ldots)$. *And at last it can be programmed into flash in the order:* $(c_1, c_4, c_3, c_2, c_6, c_5, c_9, c_8, c_7, c_{10}, c_{11}, c_{13}, c_{12}, c_{15}, c_{14} \ldots)$.

Decoding: assume we are to decode the flash cells programmed above. First read the cells sequentially and then compare c_{i+1} and c_{i+2}, for $i \geq 0$. Decode P_i as 1 if $c_{i+1} < c_{i+2}$, as 0 if $c_{i+1} > c_{i+2}$. Thus we get $(P_0 \ldots P_{15}) = (1001\ 0100\ 1110\ 1010 \ldots)$, *and by the decoding scheme in Example 8, we get* $(x_0 \ldots x_{11} \ldots) = (001\ 100\ 000\ 010 \ldots)$.

V. Lower Bound for Capacity

In this section, we present a lower bound to the capacity of the BRM code. To derive this bound, we first present a new form of rank modulation called the *star BRM*.

A. Star BRM

A Star BRM code uses $n+v$ cells. For convenience, let n be a multiple of $m-v$. v of these $n+v$ cells are called *anchors*, and we denote their cell levels by $(\ell_1, \ell_2, \cdots, \ell_v)$. The other n cells are called *storage cells*, and we denote their cell levels by $c_1, c_2, \cdots, c_n$. For $i = 1, 2, \cdots, v$, $\ell_i \in [1, D]$; for $i = 1, 2, \cdots, n$, $c_i \in [1, D]$. For $i = 1, 2, \cdots, \frac{n}{m-v}$, we define *block* B_i to be these m cell levels: $(\ell_1, \ell_2, \cdots, \ell_v, c_{(i-1)(m-v)+1}, c_{(i-1)(m-v)+2}, \cdots, c_{i(m-v)})$. We can see that these $\frac{n}{m-v}$ blocks share the same v cells, namely, the anchor cells. For $i = 1, 2, \cdots, \frac{n}{m-v}$, we require that the m cell levels in the block B_i are all different, and we use P_i to denote the permutation induced by B_i. B_i is a *realization* of P_i. Again, a

permutation sequence $(P_1, P_2, \cdots, P_{n/(m-v)})$ may have multiple realizations.

Definition 13 *(STAR BRM CODE $\mathcal{S}(n,m,D,v)$) In a Star BRM code $\mathcal{S}(n,m,D,v)$, every codeword is a permutation sequence $(P_1, P_2, \cdots, P_{n/(m-v)})$ that has at least one realization. (The meaning of the parameters n,m,D,v is as presented above.) Let $|\mathcal{S}(n,m,D,v)|$ denote the number of codewords in code $\mathcal{S}$. Then, the* capacity *of the code is*

$$cap(\mathcal{S}) = \lim_{n\to\infty} \frac{\log|\mathcal{S}(n,m,D,v)|}{n+v}.$$

To derive the capacity of Star BRM, we first show how the anchor levels $(\ell_1, \ell_2, \cdots, \ell_v)$ affect the permutation sequences. Define $Z(\ell_1, \ell_2, \cdots, \ell_v)$ as the total number of permutations that can be induced by the cell levels $(\ell_1, \ell_2, \cdots, \ell_v, c'_1, c'_2, \cdots, c'_{m-v})$, where the m cell levels are all different and all belong to the set $[1, D]$. (Here $(\ell_1, \ell_2, \cdots, \ell_v)$ are fixed, and the $m-v$ cell levels $(c'_1, c'_2, \cdots, c'_{m-v})$ can vary and therefore can have $\frac{(D-v)!}{(D-m)!}$ combinations. Some of them induce the same permutation.) It can be observed that when we permute the v anchor levels $(\ell_1, \ell_2, \cdots, \ell_v)$, the value of $Z(\ell_1, \ell_2, \cdots, \ell_v)$ remains the same. For example, when $v=3$ and $D=6$, $Z(2,3,6) = Z(3,2,6) = Z(6,2,3)$. So without loss of generality (WLOG), we assume $\ell_1 < \ell_2 < \cdots < \ell_v$.

Given $(\ell_1, \ell_2, \cdots, \ell_v)$, let $\beta(\ell_1, \ell_2, \cdots, \ell_v)$ denote the number of solutions for the variables $x_1, x_2, \cdots, x_{v+1}$ that satisfy the following two conditions: (1) $\sum_{i=1}^{v+1} x_i = m-v$; (2) $x_1 \in [0, \ell_1 - 1]$, $x_i \in [0, \ell_i - \ell_{i-1} - 1]$ for $i \in [2, v]$, and $x_{v+1} \in [0, D - \ell_v]$.

Lemma 14. *Given $D \geq m > v$, we have $Z(\ell_1, \ell_2, \cdots, \ell_v) = (m-v)! \cdot \beta(\ell_1, \ell_2, \cdots, \ell_v)$.*

Proof: Given the anchor cell levels $(\ell_1, \ell_2, \cdots, \ell_v)$, a permutation induced by $(\ell_1, \ell_2, \cdots, \ell_v, c'_1, c'_2, \cdots, c'_{m-v})$ can be uniquely determined by the following two steps: (1) determine the relative order of the $m-v$ cell levels $(c'_1, c'_2, \cdots, c'_{m-v})$ (that is, which cell level is the highest, second highest, and so on $\cdots$ among them); (2) determine how many cell levels among $(c'_1, c'_2, \cdots, c'_{m-v})$ are below ℓ_1, or between ℓ_1 and ℓ_2, or between ℓ_2 and ℓ_3, $\cdots$, or above ℓ_v. Step 1 has $(m-v)!$ choices, and step 2 has $\beta(\ell_1, \ell_2, \cdots, \ell_v)$ choices. So the conclusion holds. ∎

Lemma 15. *$Z(\ell_1, \ell_2, \cdots, \ell_v)$ is maximized when the numbers in the following set differ by at most one: $\{\ell_1 - 1, D - \ell_v\} \cup \{\ell_i - \ell_{i-1} - 1 | i = 2, 3, \cdots, v\}$. (That is, every number in the above set is either $\lfloor \frac{D-v}{v+1} \rfloor$ or $\lceil \frac{D-v}{v+1} \rceil$.)*

Proof: By Lemma 14, maximizing $Z(\ell_1, \ell_2, \cdots, \ell_v)$ is equivalent to maximizing $\beta(\ell_1, \ell_2, \cdots, \ell_v)$. Define $\alpha_1 = \ell_1 - 1$, $\alpha_i = \ell_i - \ell_{i-1} - 1$ for $i \in [2, v]$, and $\alpha_{v+1} = D - \ell_v$. Suppose there exists $i \neq j$ such that $\alpha_i \geq \alpha_j + 2$. WLOG, let $i < j$. Let $x_1, x_2, \cdots, x_{v+1}$ be variables satisfying these two conditions: (1) $\sum_{k=1}^{v+1} x_k = m - v$; (2) $x_k \in [0, \alpha_k]$ for $k \in [1, v+1]$. The number of such solutions is $\beta(\ell_1, \ell_2, \cdots, \ell_v)$. Now, let us fix the values of $x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_{j-1}, x_{j+1}, \cdots, x_{v+1}$ (in a valid solution), and see how many different values x_i can take. (Note that the value of x_j is determined by x_i.)

Let $z = D - \sum_{k \in \{1,\cdots,i-1,i+1,\cdots,j-1,j+1,\cdots,v+1\}} x_k = x_i + x_j$. Let $\gamma(z)$ denote the number of values x_i can take. The constraints are $0 \leq x_i \leq \alpha_i, 0 \leq z - x_i \leq \alpha_j$. If $z \geq \alpha_i$, $\gamma(z) = \alpha_i + \alpha_j - z + 1$; if $\alpha_j \leq z < \alpha_i$, $\gamma(z) = \alpha_j + 1$; if $z < \alpha_j$, $\gamma(z) = z + 1$. So if we increase α_j by one and decrease α_i by one, $\gamma(z)$ will not decrease although the values $\alpha_1, \alpha_2, \cdots, \alpha_{v+1}$ will become more even. So given a sequence $(\ell_1, \ell_2, \cdots, \ell_v)$, we can change it that way into a sequence that satisfies the condition in the lemma, without decreasing $\beta(\ell_1, \ell_2, \cdots, \ell_v)$. It is easy to see that when $\alpha_1, \alpha_2, \cdots, \alpha_{v+1}$ differ by at most one, no matter what their order is, $\beta(\ell_1, \ell_2, \cdots, \ell_v)$ is the same (which is the maximum value of $\beta(\ell_1, \ell_2, \cdots, \ell_v)$). ∎

Let $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$ be the v anchor levels that satisfy the condition in Lemma 15. It maximizes the value of $Z(\ell_1, \ell_2, \cdots, \ell_v)$. For convenience, we assume that $\ell_1^* < \ell_2^* < \cdots < \ell_v^*$. It is very simple to find these v values. For convenience, we use Z^* to denote $Z(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$, and use β^* to denote $\beta(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$.

The values of β^* and Z^* can be computed efficiently by the following dynamic programming algorithm of time complexity $O(D^2)$. Let $\alpha_1 = \ell_1^* - 1$, $\alpha_i = \ell_i^* - \ell_{i-1}^* - 1$ for $i \in [2, v]$, and $\alpha_{v+1} = D - \ell_v^*$. Let $w(i,j)$ denote the number of solutions for $x_1, x_2, \cdots, x_i$ such that $\sum_{k=1}^{i} x_k = j$ and $x_k \in [0, \alpha_k]$ for $k = 1, \cdots, i$. The algorithm is as follows: (1) $w(i,j) = \sum_{k=0}^{\alpha_i} w(i-1, j-k)$. Also, $w(i,j) = 0$ if $j < 0$, $w(1,j) = 1$ if $0 \leq j \leq \alpha_1$, and $w(1,j) = 0$ if $j > \alpha_1$; (2) $\beta^* = w(v+1, D-v)$, and $Z^* = (m-v)!\beta^*$.

The following theorem presents the capacity of the Star BRM.

Theorem 16. *The capacity of the Star BRM code $\mathcal{S}(n,m,D,v)$ is*

$$cap(\mathcal{S}) = \frac{\log Z^*}{m-v}.$$

Proof: We first show that $cap(\mathcal{S}) \leq \frac{\log Z^*}{m-v}$. There are $v!\binom{D}{v}$ ways to assign values to $(\ell_1, \ell_2, \cdots, \ell_v)$, which we denote by $W = \{w_1, w_2, \cdots, w_{v!\binom{D}{v}}\}$. We call $(\ell_1, \ell_2, \cdots, \ell_v, c_1, c_2, \cdots, c_n)$ the *cell-level sequence*. For $i = 1, 2, \cdots, v!\binom{D}{v}$, let $\gamma_{i,n}$ denote the

maximum set of cell-level sequences satisfying two conditions: (1) They all assign w_i to $(\ell_1, \ell_2, \cdots, \ell_v)$; (2) The permutations induced by them are all distinct.

By the definition of $Z(\ell_1, \ell_2, \cdots, \ell_v)$, every *block* can induce $Z(\ell_1, \ell_2, \cdots, \ell_v)$ permutations. Since there are $n/(m-v)$ blocks, we get $|\gamma_{i,n}| = (Z(w_i))^{\frac{n}{m-v}}$. By Lemma 15, $Z(w_i) \leq Z^*$. Since every codeword of $\mathcal{S}$ has at least one realization in some $\gamma_{i,n}$, $|\mathcal{S}(n,m,D,v)| \leq \sum_{i=1,2,\cdots,v!\binom{D}{v}} |\gamma_{i,n}| \leq v!\binom{D}{v}(Z^*)^{\frac{n}{m-v}}$.

So $cap(\mathcal{S}) = \lim_{n\to\infty} \frac{\log|\mathcal{S}(n,m,D,v)|}{n+v} \leq \lim_{n\to\infty} \frac{\log(v!\binom{D}{v}(Z^*)^{\frac{n}{m-v}})}{n} = \frac{\log Z^*}{m-v}$.

We now show that $cap(\mathcal{S}) \geq \frac{\log Z^*}{m-v}$. Say that $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*) = w_{i'}$ for some i'. Every codeword of $\mathcal{S}$ has at most one configuration in $\gamma_{i',n}$, so $|\mathcal{S}(n,m,D,v)| \geq |\gamma_{i',n}|$. So $cap(\mathcal{S}) = \lim_{n\to\infty} \frac{\log|\mathcal{S}(n,m,D,v)|}{n+v} \geq \lim_{n\to\infty} \frac{\log|\gamma_{i',n}|}{n} = \lim_{n\to\infty} \frac{\log(Z^*)^{\frac{n}{m-v}}}{n} = \frac{\log Z^*}{m-v}$. So the theorem is proved. ∎

The above proof leads to the following corollary.

Corollary 17 *The Star BRM code $\mathcal{S}(n,m,D,v)$ achieves its capacity even if the v anchor cell levels are fixed as $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$.*

The capacity of the Star BRM code $\mathcal{S}(n,m,D,v)$ is non-decreasing in D. However, when $D = (m-v+1)v + (m-v)$, the capacity reaches its maximum value. Further increasing D will not increase the capacity. That is because when $D \geq (m-v+1)v + (m-v)$, Z^* reaches its maximum value $m!/v!$.

B. Lower Bound for The Capacity of BRM

We now derive a lower bound for the capacity of the bounded rank modulation code $\mathcal{C}(n,m,D,v)$.

Theorem 18. *For the BRM code $\mathcal{C}(n,m,D,v)$, when $m \geq 2v$, its capacity*

$$cap(\mathcal{C}) \geq \frac{\log Z^* + \log v! + \log(m-2v)!}{2(m-v)}.$$

(As presented previously, Z^* is a value determined by the parameters m, D, v.)

Proof: Let $\mathcal{S}(n,m,D,v)$ be a Star BRM code with an additional constraint: every codeword of $\mathcal{S}$ has a realization in which the v anchor levels are $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$. By Corollary 17, $\mathcal{S}$ achieves capacity. For convenience, assume $n/(m-v)$ is an integer.

Let $(P_1, P_2, \cdots, P_{n/(m-v)})$ be a codeword in $\mathcal{S}$, and let $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*, c_1, c_2, \cdots, c_n) = (B_1, B_2, \cdots, B_{n/(m-v)})$ be its realization. For $i = 1, 2, \cdots, n/(m-v)$, corresponding to block B_i, we build two blocks B_i' and B_i'' of length m as follows. Say $B_i = (\ell_1^*, \ell_2^*, \cdots, \ell_v^*, c_1', c_2', \cdots, c_{m-v}')$. The first v cell levels of B_i' take values from the set $\{\ell_1^*, \ell_2^*, \cdots, \ell_v^*\}$ (we have $v!$ choices here), and the next $m-v$ cell levels of B_i' are the same as $(c_1', c_2', \cdots, c_{m-v}')$. The first v cell levels of B_i'' overlap the last v cell levels of B_i'. For the next $m-2v$ cell levels of B_i'', we first pick $m-2v \leq D-2v$ values different from the first v and the last v cell levels of B_i', then let the $m-2v$ cell levels take only those $m-2v$ values (we have $(m-2v)!$ choices here). The final v cell levels of B_i'' take values again from the set $\{\ell_1^*, \ell_2^*, \cdots, \ell_v^*\}$. Then we construct a cell-level sequence $(B_1', B_1'', B_2', B_2'', \cdots, B_{n/(m-v)}', B_{n/(m-v)}'')$, where every two adjacent blocks overlap by v. Corresponding to every codeword $s \in \mathcal{S}$, there are at least $(v!(m-2v)!)^{\frac{n}{m-v}}$ such cell-level sequences, which we denote by Q_s. It is simple to see that no two cell-level sequences in Q_s induce the same permutation sequence. On the other side, when $s \neq s'$, every pair of cell-level sequences from Q_s and $Q_{s'}$, respectively, also induce different permutation sequences. (To see that, let us call the pair of cell-level sequences q and q'. Replace all their overlapping cell levels by $(\ell_1^*, \ell_2^*, \cdots, \ell_v^*)$, and get two new cell-level sequences p and p'. The codewords s and s' are subsequences of $\mathcal{I}(p)$ and $\mathcal{I}(p')$, respectively. Since $s \neq s'$, $\mathcal{I}(p) \neq \mathcal{I}(p')$. So $\mathcal{I}(q) \neq \mathcal{I}(q')$.) We can also see that every cell-level sequence constructed above induces a codeword in the code $\mathcal{C}(2n+v,m,D,v)$.

So corresponding to the $|\mathcal{S}(n,m,D,v)|$ codewords of the Star BRM code $\mathcal{S}(n,m,D,v)$, we can find at least $|\mathcal{S}(n,m,D,v)|(v!(m-2v)!)^{\frac{n}{m-v}}$ codewords of the BRM code $\mathcal{C}(2n+v,m,D,v)$. So the capacity of code $\mathcal{C}(n,m,D,v)$ is $cap(\mathcal{C}) \geq \lim_{n\to\infty} \frac{\log|\mathcal{S}(n,m,D,v)| + (\log v! + \log(m-2v)!)\cdot\frac{n}{m-v}}{2n+v} = \lim_{n\to\infty} \frac{\log|\mathcal{S}(n,m,D,v)|}{2n} + \frac{\log v! + \log(m-2v)!}{2(m-v)} = \frac{cap(\mathcal{S})}{2} + \frac{\log v! + \log(m-2v)!}{2(m-v)} = \frac{\log Z^* + \log v! + \log(m-2v)!}{2(m-v)}$. So the theorem is proved. ∎

Corollary 19 *Let $\mathcal{C}(n,m,D,v)$ be a BRM code, and let $\mathcal{S}(n,m,D,v)$ be a Star BRM code. Then, when $m \geq 2v$,*

$$cap(\mathcal{C}) \geq \frac{1}{2} \cdot cap(\mathcal{S}).$$

In particular, when $v > 1$ or $m > 2v$, $cap(\mathcal{C}) > \frac{1}{2} \cdot cap(\mathcal{S})$.

We now present a lower bound for the case $m < 2v$. Define $A_k^n = \binom{n}{k}k! = n!/(n-k)!$, which is the number of ways to arrange k elements in n positions. Suppose $m < 2v$ and $v = k(m-v) + s$, where $k \in \mathbb{N}^+$ and $1 \leq s \leq m-v$. Let $r = m-v-s$. (So $0 \leq r \leq m-v-1$.) Define a constant $M = A_s^{m-v}(A_{m-v}^{2(m-v)-s})^{k-1}(m-v)!$. We have the following lower bound for the BRM code when $m < 2v$.

Theorem 20 *For the BRM code $\mathcal{C}(n,m,D,v)$, when $m < 2v$ and $D \geq m+r$, its capacity*

$$cap(\mathcal{C}) \geq \frac{\log(Z^* \cdot M \cdot r!)}{m+r}$$

Proof: Use the notations in Theorem 18. For each block B_i, $i = 1,2,\cdots,n/(m-v)$, we first construct $k+2$ blocks $B_i^{(1)}, B_i^{(2)}, \cdots, B_i^{(k+2)}$. And then build a cell-level sequence $q = (B_1^{(1)}, B_1^{(2)}, \cdots, B_1^{(k+2)}, \cdots, B_{n/(m-v)}^{(1)}, B_{n/(m-v)}^{(2)}, \cdots, B_{n/(m-v)}^{(k+2)})$ of length $n' = (k+2)n + v = \frac{n(m+r)}{m-v} + v$, such that every two adjacent blocks overlap by v. Define $B_0^{(k+2)} = (1,2,\cdots,m-v,\ell_1^*,\ell_l^*,\cdots,\ell_v^*)$. Then the first v cell levels of $B_i^{(1)}$ are the same as the last v cell levels in $B_{i-1}^{(k+2)}$, $\forall\ i = 1,\cdots,n/(m-v)$, and the first v cell levels of $B_i^{(j)}$ are exactly the last v cell levels of $B_i^{(j-1)}$, $\forall\ j = 2,\cdots,k+2$. So we are left to build the last $m-v$ cell levels of $B_i^{(j)}$. Assign $(c_1',c_2',\cdots,c_{m-v}')$ to the last $m-v$ cell levels of $B_i^{(1)}$. For $B_i^{(2)}$, the $(v+1)$-th through the $(v+r)$-th cell levels take $r \leq D-m$ numbers from $[1,D]$ that are different from B_i. We have $r!$ choices here. And the last s cell levels are assigned s values from $\{\ell_1^*,\ell_l^*,\cdots,\ell_v^*\}$ that are different from the last $(k-1)(m-v)+s$ cell levels of $B_{i-1}^{(k+2)}$. Thus identical levels in a block are avoided and we have A_s^{m-v} choices here. For the last $m-v$ cell levels of $B_i^{(j)}$, $j = 3,\ldots,k+1$, we pick $m-v$ values from $\{\ell_1^*,\ell_l^*,\cdots,\ell_v^*\}$ that are not in the last $(m-v)(j-3)+s$ cell levels of $B_i^{(j-1)}$ nor in the last $(k-j+1)(m-v)+s$ cell levels of $B_{i-1}^{(k+2)}$. There are $A_{m-v}^{2(m-v)-s}$ choices for each $j = 3,\cdots,k+1$. Finally, the last $m-v$ cell levels of $B_i^{(k+2)}$ are chosen from $\{\ell_1^*,\ell_l^*,\cdots,\ell_v^*\}$ such that they are different from the last $(m-v)(k-1)+s$ cell levels of $B_i^{(k+1)}$, which results in $(m-v)!$ choices.

Notice q is a valid cell-level sequence of $\mathcal{C}(n',m,D,v)$ as each cell level is no more than D and any block in q has m different levels. For each codeword s in $\mathcal{S}$, there are at least $M \cdot r!$ such cell-level sequences, denoted by Q_s. Also notice that similar to Theorem 18, neither two cell-level sequences in Q_s nor two cell-level sequences in distinct Q_s and $Q_{s'}$ induce the same permutation sequence for $\mathcal{C}(n',m,D,v)$. Moreover, the number of distinct permutation sequences in $\mathcal{C}(n',m,D,v)$ is at least $\sum_{s\in\mathcal{S}}|Q_s| \geq |\mathcal{S}(n,m,D,v)|(M\cdot r!)^{\frac{n}{m-v}}$. Therefore, $cap(\mathcal{C}) \geq lim_{n\to\infty}\frac{\log|\mathcal{S}(n,m,D,v)|(M\cdot r!)^{\frac{n}{m-v}}}{n'} = lim_{n\to\infty}\frac{(m-v)\log|\mathcal{S}(n,m,D,v)|}{n(m+r)} + \frac{\log(M\cdot r!)}{m+r} = \frac{(m-v)cap(\mathcal{S})}{m+r} + \frac{\log(M\cdot r!)}{m+r} = \frac{\log(Z^*\cdot M\cdot r!)}{m+r}$. Thus we have proved the theorem. ∎

Corollary 21 *Let $\mathcal{C}(n,m,D,v)$ be a BRM code, and $\mathcal{S}(n,m,D,v)$ be a Star BRM code. Then if $m < 2v$ and $D \geq m+r$,*

$$cap(\mathcal{C}) > \frac{(m-v)cap(\mathcal{S})}{m+r} = \frac{cap(\mathcal{S})}{k+2}$$

VI. Concluding Remarks

The question of what overlap provides the highest capacity for a given permutation size and a given maximum level is partially discussed in this paper. Denote this optimal overlap by $v^*(D)$. The following observations show the two extreme cases and the result of Theorem 5:

1) When $D = m$, $cap(\mathcal{C}(n,m,m,v)) = \frac{log(m-v)!}{m-v}$. Therefore, $cap(\mathcal{C}(n,m,m,0)) > cap(\mathcal{C}(n,m,m,1)) > \cdots > cap(\mathcal{C}(n,m,m,m-1))$. We can conclude that $v^*(m) = 0$.
2) When $D = \infty$, it is clear that $cap(\mathcal{C}(n,m,\infty,v)) = \frac{\log(m!/v!)}{m-v}$. Then $cap(\mathcal{C}(n,m,\infty,0)) < cap(\mathcal{C}(n,m,\infty,1)) < \cdots < cap(\mathcal{C}(n,m,\infty,m-1))$, which implies that $v^*(\infty) = m-1$
3) When $D \geq m+2$, $cap(\mathcal{C}(n,m,D,1)) > cap(\mathcal{C}(n,m,D,0))$. So the optimal overlap for $D \geq m+2$ satisfies $v^*(D) \geq 1$.

The exact optimal overlap values for $m+1 \leq D < \infty$ are not thoroughly examined, which can be our future work direction. Besides, for any rate no more than the capacity, the exact forms of the permutation encoders and decoders in addition to their efficiency and complexities are still left to be worked on.

In addition, a generalized BRM code can be viewed as a set of n cells, among which we choose subsets of size m and form permutations. All cell levels are no more than D and two subsets may overlap. Under this framework, how to make choices of the subsets so as to optimize the capacity is still an open problem.

In summery, this paper used the tools of labeled graphs to find the capacities of BRM codes with one overlap. In particular, it showed that if two extra charge levels are given, one can use them by way of overlap and achieve higher capacity than non-overlap codes. In addition, star BRM code is introduced to obtain a lower bound for the capacity of BRM codes.

References

[1] V. Bohossian, A. Jiang and J. Bruck, "Buffer codes for asymmetric multi-level memory," *Proc. IEEE ISIT*, 2007, pp. 1186-1190.

[2] J. E. Brewer and M. Gill, *Nonvolatile memory technologies with emphasis on flash*, Chapter 12, Wiley-IEEE, 2007.

[3] Y. Cassuto, M. Schwartz, V. Bohossian and J. Bruck, "Codes for asymmetric limited-magnitude errors with application to multi-level flash memories," *Proc. IEEE ISIT*, 2007.

[4] H. Finucane, Z. Liu and M. Mitzenmacher, "Designing floating codes for expected performance," *Proc. of the 46th Annual Allerton Conference*, 2008.

[5] A. Jiang, V. Bohossian and J. Bruck, "Floating codes for joint information storage in write asymmetric memories," *Proc. IEEE ISIT*, 2007, pp. 1166-1170.
[6] A. Jiang and J. Bruck, "Joint coding for flash memory storage," *Proc. IEEE ISIT*, 2008, pp. 1741-1745.
[7] A. Jiang, R. Mateescu, M. Schwartz and J. Bruck, "Rank modulation for flash memories," *Proc. IEEE ISIT*, 2008, pp. 1731-1735.
[8] A. Jiang, M. Schwartz and J. Bruck, "Error-correcting codes for rank modulation," *Proc. IEEE ISIT*, 2008, pp. 1736-1740.
[9] B. H. Marcus, R. M. Roth and P. H. Siegel, *An introduction to coding for constrained systems*, 5th Edition, 2001, available at $http://www.math.ubc.ca/\ marcus/Handbook/index.html$.
[10] E. Yaakobi, A. Vardy, P. H. Siegel and J. Wolf, "Multidimensional flash codes," *Proc. of the 46th Annual Allerton Conference*, 2008.