Variable-level Cells for Nonvolatile Memories

Anxiao (Andrew) Jiang
Computer Science and Eng. Dept.
Texas A&M University
College Station, TX 77843
ajiang@cse.tamu.edu

Hongchao Zhou
Electrical Engineering Dept.
California Institute of Technology
Pasadena, CA 91125
hzhou@caltech.edu

Jehoshua Bruck
Electrical Engineering Dept.
California Institute of Technology
Pasadena, CA 91125
bruck@caltech.edu

***Abstract*—For many nonvolatile memories, – including flash memories, phase-change memories, *etc.*, – maximizing the storage capacity is a key challenge. The existing method is to use multi-level cells (MLC) of more and more levels. The number of levels supported by MLC is seriously constrained by the worst-case performance of cell-programming noise and cell heterogeneity.**

In this paper, we present *variable-level cells (VLC)*, a new scheme for maximum storage capacity. It adaptively chooses the number of levels and the placement of the levels based on the actual programming performance. We derive its storage capacity, and present an optimal data representation scheme. We also study rewriting schemes for VLC, and present inner and outer bounds to its capacity region.

I. Introduction

For nonvolatile memories (NVMs) – including flash memories, phase-change memories (PCMs), memristors, *etc.*, – maximizing the storage capacity is a key challenge. The existing method is to use multi-level cells (MLCs) of more and more levels, where a cell of q discrete levels can store $\log_2 q$ bits [1]. Flash memories with 4 and 8 levels have been used in products, and MLCs with 16 levels have been demonstrated in prototypes. For PCMs, cells with 4 or more levels have been in development. How to maximize the number of levels in cells is a most important topic for study.

The number of levels that can be programmed into cells is seriously constrained by the noise in cell programming and by cell heterogeneity [1]. We explain it with flash memories as an example, and the concepts can be naturally extended to PCMs and memristors. A flash memory uses the charge stored in floating-gate cells to store data, where the amount of charge in a cell is quantized into q values to represent q discrete levels. Cell programming – the operation of injecting charge into cells – is a noisy process, which means that the actual increase in the cell levels can deviate substantially from the target value. And due to the block erasure property, – which means that to remove charge from any cell, a whole block of about 10^5 cells must be erased together to remove all their charge, – during the writing procedure, the cell levels are only allowed to monotonically increase using charge injection. That makes it infeasible to correct over-injection errors [1]. Beside cell-programming noise, the difficulty in programming is also caused by cell heterogeneity, which means that even when the same voltage is used to program different cells, the increments in the different cells' levels can differ substantially, due to the heterogeneity in cell material and geometry [8]. Since memories use parallel programming for high write speed, a common voltage is used to program many cells during a programming step, which cannot be adjusted for individual cells [1], [8]. As cell sizes scale down, the cell heterogeneity will be even more significant [1].

The storage capacity of MLC is limited by the worst-case performance of cell-programming noise and cell heterogeneity [1], [8]. We illustrate it in Fig. 1 (a). A *safety gap* is needed to separate two adjacent levels to prevent errors after programming. The charge level for an individual cell has a random distribution due to the cell-programming noise [1], [8]. The actual value of the charge level varies from one write to another. Due to cell heterogeneity, the charge-level distributions of different cells in the same level shift away from each other, which widens the overall charge-level distribution of the level [1], [8]. Since MLC uses fixed levels for storage, it needs to accommodate the worst-case programming performance: the charge-level range for a level is set to be sufficiently wide to accommodate not only the worst-case programming noise for each cell, but also the worst-case cell heterogeneity. That limits the number of levels in MLC.

In this paper, we introduce a new storage scheme named *variable-level cells (VLC)* for maximum storage capacity. It has two unique properties: the number of levels is not fixed, and the positions of the levels are chosen adaptively during programming. More specifically, we program the levels sequentially from low to high. After level i is programmed, we program level $i+1$ such that the gap between the two adjacent levels is at least the required safety gap. (There are many ways to differentiate the cells in different levels. For example, we can require the cells of the same level to have charge levels within δ from each other, and require cells in different levels to have charge levels at least Δ away from each other, for appropriately chosen parameters δ, Δ.) We program as many levels into the cells as possible until the highest programmed level reaches the physical limit.

The VLC scheme places the levels as compactly as possible, and maximizes the number of programmed levels, which is determined by the *actual* instead of the *worst-case* programming performance. It is illustrated in Fig. 1 (b). Note that for a set of cells programmed in parallel, their heterogeneity is usually not as significant as the worst-case heterogeneity of all memory cells, which helps narrow the actual charge-level range for a level [1]. Furthermore, the actual cell-programming noise is often not as large as its worst-case value, which further narrows the actual range of charge levels for the level. The VLC scheme places level $i+1$ as low as possible based on the actual position of level i. The better the actual programming

978-1-4577-0594-6/11/$26.00 ©2011 IEEE

performance is, the more levels we write into the cells.

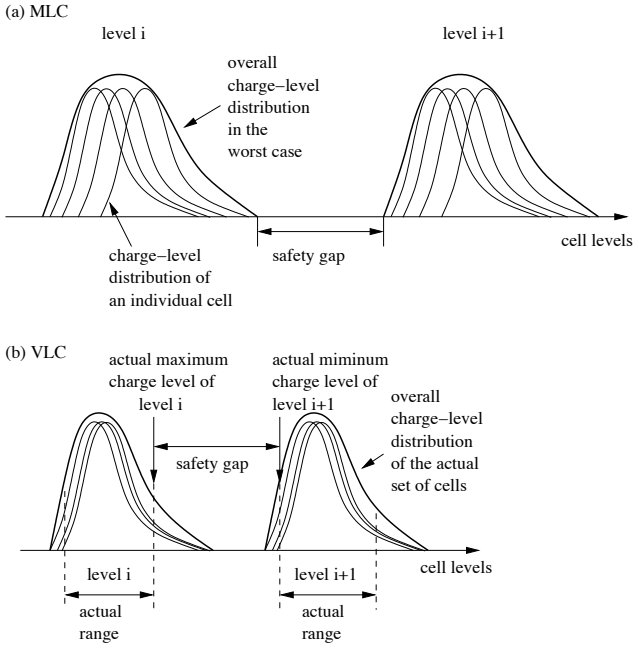

Fig. 1. Charge-level distribution of (a) MLC; (b) VLC.

The VLC scheme shifts data representation into the stochastic regime, because the number of levels actually used is not determined in advance. New coding schemes are needed for this new paradigm. In this paper, we present a data representation scheme, and prove that it achieves the storage capacity of VLC. We also study rewriting codes, which are important for improving the longevity of flash memories and PCMs [5], [7], and present bounds for achievable rates.

Due to the space limitation, we present the proofs of a number of lemmas and theorems in [6].

II. Data Representation and Capacity of VLC

In this section, we present a probabilistic model for VLC, study its representation scheme, and derive its capacity.

A. Discrete Model for VLC

For a storage scheme, it is key to have a discrete model that not only enables efficient code designs, but is also robust to the physical implementation of the scheme. In this paper, we use the following simple probabilistic model for VLC.

Let q denote the maximum number of levels we can program into cells, and call the q levels level 0, level 1, $\cdots$, level $q-1$. Let n denote the number of cells, and for $i = 1, 2, \cdots, n$, denote the level of the ith cell by $c_i \in \{0, 1, \cdots, q-1\}$. Before writing, all cells are at level 0. Let $L = (\ell_1, \ell_2, \cdots, \ell_n) \in \{0, 1, \cdots, q-1\}^n$ denote the *target levels*, which means that for $i = 1, \cdots, n$, we plan to program c_i as ℓ_i. [1] To program cells to the target levels L, we first program level 1 (namely, push some cells from level 0 to level 1), then program level 2, level 3, $\cdots$, until we reach a certain level i such that its charge levels are so close to the physical limit that we will not be able to program level $i+1$. All the cells that should belong to levels $1, 2, \cdots, i$ are successfully programmed to those levels. The cells that should belong to levels $\{i+1, i+2, \cdots, \max_{1\le j\le n} \ell_j\}$ are still in level 0 (together with the cells that should belong to level 0). So the final cell levels are $L_i \triangleq (c'_1, c'_2, \cdots, c'_n)$, where for $j = 1, \cdots, n$, $c'_j = \ell_j$ if $1 \le \ell_j \le i$, and $c'_j = 0$ otherwise.

For $i = 1, 2, \cdots, q-1$, let p_i denote the probability that level i can be programmed *given* that levels $1, 2, \cdots, i-1$ are successfully programmed. (And for convenience, define $p_q = 0$.) Let T denote the target levels, and S denote the *written* levels. So when $T = L \in \{0, 1, \cdots, q-1\}^n$, for $i = 0, 1, \cdots, q-1$, we have $Pr\{S = L_i\} = (1 - p_{i+1}) \prod_{j=1}^{i} p_j$.

We define the *capacity of VLC* by

$$C = \lim_{n\to\infty} \frac{1}{n} \max_{P_T(t)} I(T; S),$$

where $P_T(t)$ is the probability distribution of T, and $I(T; S)$ is the mutual information of T and S. [2]

B. Data Representation Schemes

We present a data representation scheme with a nice property: Every level i (for $i = 1, 2, \cdots, q-1$) encodes a separate set of information bits. It enables efficient encoding and decoding of data. The code also achieves capacity and is therefore optimal. The code is of *constant weight*: The number of cells assigned to each level is fixed for all codewords.

Let $\mu_1, \mu_2, \cdots, \mu_{q-1} \in (0, 1)$ be parameters. The codewords of our code are the target levels T that have this property: "$n\mu_1$ cells are assigned to level 1; and for $i = 2, 3, \cdots, q-1$, $n\mu_i \prod_{j=1}^{i-1} (1 - \mu_j)$ cells are assigned to level i." (This is a general definition of constant-weight codes. Clearly, μ_i denotes the number of cells assigned to level i divided by the number of cells assigned to levels $\{0, i, i+1, \cdots, q-1\}$. Here we consider $n \to \infty$ and $p_i > 0$ for $1 \le i \le q-1$.) The constant-weight code enables convenient encoding and decoding methods as follows. Since there are $\binom{n}{n\mu_1}$ ways to choose the $n\mu_1$ cells in level 1, level 1 can encode $\log_2 \binom{n}{n\mu_1} \doteq nH(\mu_1)$ information bits. Then, for $i = 2, 3, \cdots, q-1$, given the cells already assigned to levels $\{1, 2, \cdots, i-1\}$, there are $\binom{n\prod_{j=1}^{i-1}(1-\mu_j)}{n\mu_i \prod_{j=1}^{i-1}(1-\mu_j)}$ ways to choose the $n\mu_i \prod_{j=1}^{i-1} (1 - \mu_j)$ cells in level i; so level i can encode $\log_2 \binom{n\prod_{j=1}^{i-1}(1-\mu_j)}{n\mu_i \prod_{j=1}^{i-1}(1-\mu_j)} \doteq \left(n \prod_{j=1}^{i-1} (1 - \mu_j)\right) H(\mu_i)$ information bits. The mapping from cells in level i to information bits that level i represents has a well-studied solution in enumerative source coding [2], so we skip its details.

Given a stream of information bits, we can store its first $nH(\mu_1)$ bits in level 1, its next $n(1-\mu_1)H(\mu_2)$ bits in level

[1] Since VLC uses the relative positions of charge levels to store data, we usually require for $i = 0, 1, \cdots, \max_{1\le j\le n} \ell_j$, at least one cell is assigned to level i. However when $n \to \infty$, this constraint has a negligible effect on the code rate. So when we analyze capacity, this constraint can be neglected.

[2] Here we view the n cells as *one* symbol for the channel, and normalize its capacity by the number of cells. The capacity defined this way equals the *expected* number of bits a cell can store.

2, its next $n(1-\mu_1)(1-\mu_2)H(\mu_3)$ bits in level 3, and so on. This makes encoding and decoding convenient despite the nondeterministic behavior of writing. The *expected* number of information bits that can be written into the n cells is $\sum_{i=1}^{q-1}\left(\prod_{j=1}^{i}p_j\right)\left(n\prod_{j=1}^{i-1}(1-\mu_j)\right)H(\mu_i)$. So the *rate* of the code, measured as number of stored bits per cell, is

$$R=\sum_{i=1}^{q-1}\left(\prod_{j=1}^{i}p_j\right)\left(\prod_{j=1}^{i-1}(1-\mu_j)\right)H(\mu_i)$$

Let us define $A_1, A_2, \cdots, A_{q-1}$ recursively: $A_{q-1}=2^{p_{q-1}}$; and for $i=q-2, q-3, \cdots, 1$, $A_i=(1+A_{i+1})^{p_i}$. Theorem 2 below shows the maximum rate of the code and the corresponding optimal configuration of the parameters $\mu_1, \mu_2, \cdots, \mu_{q-1}$. We first prove the following lemma.

Lemma 1. *Let $x\in[0,1]$ and $y\in[0,1]$ be given numbers. Let $\mu^*=\frac{1}{1+2^{\frac{y}{x}}}$. Then $\max_{\mu\in[0,1]} xH(\mu)+y(1-\mu)=xH(\mu^*)+y(1-\mu^*)=\log_2\left(1+2^{\frac{y}{x}}\right)^x$.*

Proof: Define $f(\mu)\triangleq xH(\mu)+y(1-\mu)$. Then $f(\mu)=y-\frac{1}{\ln 2}(x\mu\ln\mu+x(1-\mu)\ln(1-\mu)+y\mu\ln 2)$. So $f'(\mu)=-\frac{1}{\ln 2}\left(x\ln\frac{\mu}{1-\mu}+y\ln 2\right)$, where $f'(\mu)$ is the derivative of $f(\mu)$. By setting $f'(\mu)=0$, we get $\mu=\frac{1}{1+2^{\frac{y}{x}}}\triangleq\mu^*$. And we get $f(\mu^*)=\log_2\left(1+2^{\frac{y}{x}}\right)^x$. ∎

Theorem 2. *The maximum rate of the constant-weight code is*

$$R=\log_2 A_1,$$

which is achieved when $\mu_i=A_i^{-\frac{1}{p_i}}$ for $i=1,2,\cdots,q-2$ and $\mu_{q-1}=\frac{1}{2}$.

Example 3. *Consider VLC constant-weight codes with $q=5$. We have $A_4=2^{p_4}$, $A_3=(1+2^{p_4})^{p_3}$, $A_2=(1+(1+2^{p_4})^{p_3})^{p_2}$, $A_1=\left(1+(1+(1+2^{p_4})^{p_3})^{p_2}\right)^{p_1}$.*

By Theorem 2, to maximize the rate of the code, we should choose the parameters μ_1,μ_2,μ_3,μ_4 as follows:

$$\mu_1=\frac{1}{1+(1+(1+2^{p_4})^{p_3})^{p_2}},\quad \mu_2=\frac{1}{1+(1+2^{p_4})^{p_3}},$$
$$\mu_3=\frac{1}{1+2^{p_4}},\quad \mu_4=\frac{1}{2}.$$

The above parameters make the code achieve the maximum rate $R=\log_2\left(1+(1+(1+2^{p_4})^{p_3})^{p_2}\right)^{p_1}$. □

We now discuss briefly data representation for VLC when n is small. In this case, it can be beneficial to use codes that are not of constant weight to improve code rates. At the same time, the need for every target level to contain at least one cell no longer has a negligible effect on the code rates. We illustrate such codes with the following example.

Example 4. *Consider $n=4$ cells that can have at most $q=3$ levels. We show a code in Fig. 2, which stores 3 information bits in level 1 and 1 information bit in level 2. (The four numbers inside a box are the cell levels, and the bold-font numbers beside a box are the corresponding information bits.) Even if only level 1 and not level 2 can be programmed, we can still store 3 bits. The rate of the code is $3p_1+p_1p_2$ bits per cell.* □

C. Capacity of VLC

We now derive the capacity of VLC, and prove that the constant-weight code shown above is optimal.

We first present a channel model for a single cell. Let X denote the target level for a cell, and let Y denote the actual state of the cell after writing. Clearly, $X\in\{0,1,\cdots,q-1\}$. The level X can be successfully programmed with probability $p_1p_2\cdots p_X$ if $X\geq 1$, and with probability $p_1p_2\cdots p_{q-1}$ if $X=0$; and if so, we get $Y=X$. It is also possible that level X is not successfully programmed. For $i=0,1,\cdots,q-2$, the highest programmed level will be level i with probability $(1-p_{i+1})\prod_{j=1}^{i}p_j$; and if so, the cells with target levels in $\{0,i+1,i+2,\cdots,q-1\}$ will all remain in level 0. In that case, if $X=0$ or $i+1\leq X\leq q-1$, we denote that state of the cell after writing (namely, Y) by $E_{\{0,i+1,i+2,\cdots,q-1\}}$ and call it a *partial erasure*, because it is infeasible to tell which level in $\{0,i+1,i+2,\cdots,q-1\}$ is the target level of the cell. So we have $Y\in\{0,1,\cdots,q-1\}\cup\{E_{\{0,1,2,\cdots,q-1\}},E_{\{0,2,3,\cdots,q-1\}},\cdots,E_{\{0,q-1\}}\}$. We call the channel the *partial-erasure channel*. Examples of the channel for $q=2,3$ are shown in Fig. 3, where the states in rectangles are the partial erasures. (We can see that when $q=2$, the channel is the same as the binary erasure channel (BEC) with erasure probability $1-p_1$.)

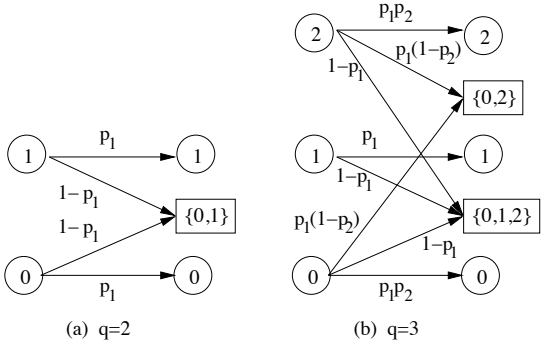

Fig. 3. Partial-erasure channel for q levels. (a) $q=2$. (b) $q=3$.

Lemma 5. *The capacity of the partial-erasure channel for q levels is $\log_2 A_1$ bits per cell.*

Theorem 6. *The capacity of VLC is*

$$C=\log_2 A_1.$$

The above theorem shows that the constant-weight code introduced in the previous subsection achieves capacity.

III. Rewriting Data in VLC

In this section, we study codes for rewriting data in VLC, and bound its achievable rates. There has been extensive study on rewriting codes for flash memories and PCMs (for both single-level cells (SLCs) and MLCs) for achieving longer memory lifetime [5], [7]. In the well known write-once memory (WOM) model, the cell levels can only increase when data

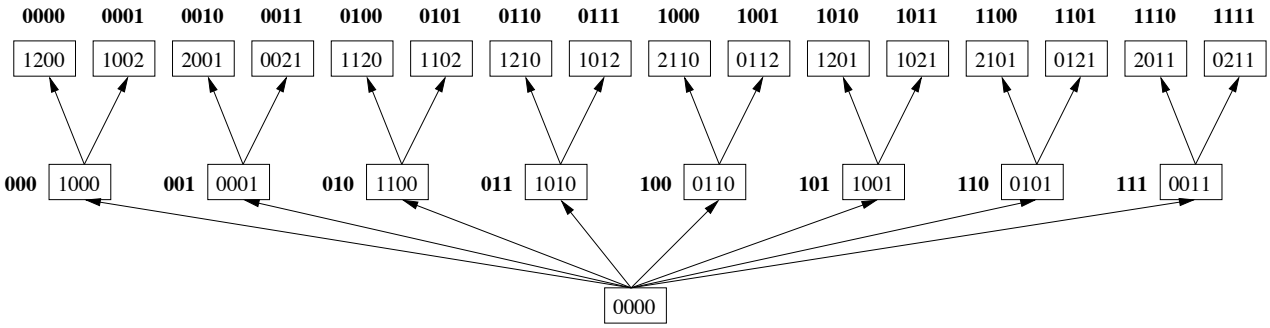

Fig. 2. A VLC code with $n = 4$ cells and $q = 3$ levels that stores 3 bits in level 1 and 1 bit in level 2.

are rewritten [4]. For flash memories and PCMs, the model describes the behavior of cells between two global erasure operations. Since erasures reduce the quality of cells, it is highly desirable to avoid them. Given the number of rewrites, T, our objective is to maximize the rates of the code for the T rewrites, when cell levels can only increase for rewriting.

A. Codes for Rewriting Data

We first consider some specific code constructions. Consider a VLC cell group that has n cells of q levels. Let $p_1, p_2, \cdots, p_{q-1}$ be the same probabilities as defined before. And for convenience, we define $p_q = 0$.

Example 7. PARITY CODE FOR REWRITING IN VLC

Let $(c_1, c_2, \cdots, c_n) \in \{0, 1, \cdots, q-1\}^n$ denote the n cells' levels. Let them represent n bits of data, $(b_1, b_2, \cdots, b_n) \in \{0,1\}^n$ this way: for $1 \le i \le n$, $b_i = c_i \mod 2$. (For convenience, we assume $n \to \infty$, and we have q auxiliary cells with target levels $0, 1, \cdots, q-1$, respectively. The auxiliary cells will ensure every programmed level will maintain at least one cell, and will help us tell the levels of the n cells.) Clearly, for every rewrite, a cell's level needs to increase by at most one. The rewriting has to end when we cannot program a higher level. The rate of the code is one bit per cell for each rewrite. And the expected number of rewrites this parity code can support is $\sum_{i=1}^{q-1} i \cdot (p_1 p_2 \cdots p_i (1 - p_{i+1})) = p_1 (1 + p_2 (1 + p_3 (\cdots + p_{q-2} (1 + p_{q-1}))))$. □

More generally, given a WOM code that rewrites k bits of data t times in n two-level cells [4], by a similar level-by-level approach, we can get a rewriting code in VLC of rate k/n that supports $tp_1 (1 + p_2 (1 + p_3 (\cdots + p_{q-2} (1 + p_{q-1}))))$ rewrites in expectation.

B. Bounding the Capacity Region for Rewriting in VLC

We now study the achievable rates for rewriting in VLC. Note that unlike MLC, which are deterministic, the highest programmable level of a VLC group is a random variable. So we need to define code rates accordingly.

Consider a VLC group of n cells, whose highest programmable level is a random variable $h \in \{1, 2, \cdots, q-1\}$. (We assume $h \ge 1$ – namely $p_1 = 1$ – for the convenience of presentation. The analysis can be extended to $h \ge 0$.) Note that the value of h remains unknown until level h is programmed. To simplify rate analysis, we suppose that there are q auxiliary cells $a_0, a_1, \cdots, a_{q-1}$ in the same VLC group, whose target levels are $0, 1, \cdots, q-1$, respectively. For $i = 1, \cdots, h$, when level i is programmed, the auxiliary cell a_i will be raised to level i and always remain there. If $h < q-1$, after level h is programmed (at which point we find that level $h+1$ cannot be programmed), we push $a_{h+1}, \cdots, a_{q-1}$ to level h, too. So having more than one auxiliary cell in a level i indicates $h = i$. For sufficiently large n, the q auxiliary cells have a negligible effect on the code rate.

Now consider N VLC groups $G_1, G_2, \cdots, G_N$, each of n cells. (For capacity analysis, we consider $N \to \infty$.) For $i = 1, \cdots, N$, denote the highest programmable level of G_i by $h_i \in \{1, \cdots, q-1\}$, and denote its cells by $(c_{i,1}, \cdots, c_{i,n})$. Here $h_1, \cdots, h_N$ are i.i.d. random variables, where for $1 \le i \le N$ and $1 \le j \le q-1$, $Pr\{h_i = j\} = p_1 p_2 \cdots p_j (1 - p_{j+1})$. (Note $p_1 = 1$ and $p_q \triangleq 0$.) If the *target level* of cell $c_{i,j}$ is $\ell_{i,j}$, we will program it to level $\min\{\ell_{i,j}, h_i\}$. Then if $h_i < q-1$ and the *written level* of cell $c_{i,j}$ is h_i, we say that the cell is in the *partially-erased state* E_{h_i}, since its target level could be any value in $\{h_i, h_i + 1, \cdots, q-1\}$. In addition, for any two vectors $\mathbf{x} = (x_1, x_2, \cdots, x_k)$ and $\mathbf{y} = (y_1, y_2, \cdots, y_k)$, we say $\mathbf{x} \le \mathbf{y}$ if $x_i \le y_i$ for $i = 1, \cdots, k$.

Definition 8. *A $(T, V_1, V_2, \cdots, V_T)$ rewriting code for the N VLC groups consists of T pairs of encoding and decoding functions $\{(f_t, g_t)\}_{t=1}^T$, with the message index sets $I_t = \{1, 2, \cdots, V_t\}$, the encoding functions $f_t : I_t \times \{0, 1, \cdots, q-1\}^{Nn} \to \{0, 1, \cdots, q-1\}^{Nn}$, and the decoding functions $g_t : \{0, 1, \cdots, q-1\}^{Nn} \to I_t$. Let $\mathbf{x}_0^{Nn} = (0, 0, \cdots, 0) \in \{0, 1 \cdots, q-1\}^{Nn}$. Given any sequence of T messages $m_1 \in I_1, m_2 \in I_2, \cdots, m_T \in I_T$, for the T rewrites, the target levels for the cells $(c_{1,1}, \cdots, c_{1,n}, c_{2,1}, \cdots, c_{2,n}, \cdots\cdots, c_{N,1}, \cdots, c_{N,n})$ are $\mathbf{x}_1^{Nn} = f_1(m_1, \mathbf{x}_0^{Nn})$, $\mathbf{x}_2^{Nn} = f_2(m_2, \mathbf{x}_1^{Nn})$, $\cdots$, $\mathbf{x}_T^{Nn} = f_T(m_T, \mathbf{x}_{T-1}^{Nn})$, respectively, where $\mathbf{x}_{t-1}^{Nn} \le \mathbf{x}_t^{Nn}$ for $t = 1, \cdots, T$. However, while the* target cell levels *for the tth rewrite (for $t = 1, \cdots, T$) are $\mathbf{x}_t^{Nn} = (\ell_{1,1}, \cdots, \ell_{1,n}, \ell_{2,1}, \cdots, \ell_{2,n}, \cdots\cdots, \ell_{N,1}, \cdots, \ell_{N,n})$, the* written cell levels *are $\mathbf{y}_t^{Nn} = (\ell'_{1,1}, \cdots, \ell'_{1,n}, \ell'_{2,1}, \cdots, \ell'_{2,n}, \cdots\cdots, \ell'_{N,1}, \cdots, \ell'_{N,n})$,*

where $\ell'_{i,j} = \min\{\ell_{i,j}, h_i\}$. *For decoding, it is required that for* $t = 1, \cdots, T$, *we have* $Pr\{g_t\left(\mathbf{y}_t^{Nn}\right) = m_t\} \to 1$ *as* $N \to \infty$.

For $t = 1, \cdots, T$, *define* $R_t = \frac{1}{Nn}\log_2 V_t$. *Then* $(R_1, R_2, \cdots, R_T)$ *is called the rate vector of the code.* □

We call the closure of the set of all rate vectors the *capacity region*, and denote it by $\mathcal{A}_T$. We present its inner/outer bounds.

1) Inner Bound to Capacity Region: We consider a *sub-channel code* for VLC. Let $c_1, c_2, \cdots, c_N$ be N cells, one from each of the N VLC groups. (The Nn cells in the N VLC groups can be partitioned into n such "sub-channels.") We define the rewriting code for the N cells in the same way as in Definition 8 (by letting $n = 1$). We denote its capacity region by $\tilde{\mathcal{A}}_T$. Clearly, for any given n, we have $\tilde{\mathcal{A}}_T \subseteq \mathcal{A}_T$.

Let $\mathcal{L} = \{0, 1, \cdots, q-1\}$ denote the set of target levels. Let $\mathcal{E} = \{E_1, E_2, \cdots, E_{q-2}\}$ denote the set of partially-erased states. Then $\mathcal{L} \cup \mathcal{E}$ are written levels. For two random variables X, Y taking values in $\mathcal{L}$, we say "$X \Rightarrow Y$" if $Pr\{X = x, Y = y\} = 0$ for any $0 \le y < x \le q-1$. Let random variables $S_1, S_2, \cdots, S_T$ form a Markov chain that takes values in $\mathcal{L}$. We say "$S_1 \Rightarrow S_2 \Rightarrow \cdots \Rightarrow S_T$" if $S_{t-1} \Rightarrow S_t$ for $t = 2, 3, \cdots, T$. For $i = 1, 2, \cdots, T$, let $(s_{i,0}, s_{i,1}, \cdots, s_{i,q-1})$ denote the probability distribution where $s_{i,j} = Pr\{S_i = j\}$ for $j = 0, 1, \cdots, q-1$.

Given the random variables $S_1, S_2, \cdots, S_T$, we define $\alpha_{i,j}$ and $B_{i,j}$ (for $i = 1, 2, \cdots, T$ and $j = 1, 2, \cdots, q-2$) as follows. Let $\alpha_{i,j} = \left(\sum_{k=j}^{q-1} s_{i,k}\right)\left(\prod_{k=2}^{j} p_k\right)(1 - p_{j+1})$. We define $B_{i,j}$ to be a random variable taking values in $\{j, j+1, \cdots, q-1\}$, where $Pr\{B_{i,j} = k\} = s_{i,k} / \left(\sum_{\ell=j}^{q-1} s_{i,\ell}\right)$ for $k = j, j+1, \cdots, q-1$. We now present an inner bound to $\tilde{\mathcal{A}}_T$. Since $\tilde{\mathcal{A}}_T \subseteq \mathcal{A}_T$, it is also an inner bound to $\mathcal{A}_T$.

Theorem 9. *Define* $\mathcal{D}_T = \{(R_1, R_2, \cdots, R_T) \in \mathbb{R}^T \mid$ *there exist Markov-chain random variables* $S_1, S_2, \cdots, S_T$ *taking values in* $\{0, 1, \cdots, q-1\}$, *such that* $S_1 \Rightarrow S_2 \Rightarrow \cdots \Rightarrow S_T$ *and*

$$\begin{aligned} R_1 \le\ & H(S_1) - \textstyle\sum_{i=1}^{q-2} \alpha_{1,i} H(B_{1,i}), \\ R_2 \le\ & H(S_2|S_1) - \textstyle\sum_{i=1}^{q-2} \alpha_{2,i} H(B_{2,i}), \\ & \vdots \\ R_T \le\ & H(S_T|S_{T-1}) - \textstyle\sum_{i=1}^{q-2} \alpha_{T,i} H(B_{T,i}).\} \end{aligned}$$

Then, we have $\mathcal{D}_T \subseteq \tilde{\mathcal{A}}_T$.

Note that if $p_2 = p_3 = \cdots = p_{q-1} = 1$ (namely, every cell can be programmed to the highest level $q-1$ with guarantee), we get $\alpha_{i,j} = 0$ for all i, j. Consequently, the set of achievable rates presented in the above theorem, $\mathcal{D}_T$, becomes $\mathcal{D}_T = \{(R_1, R_2, \cdots, R_T) \in \mathbb{R}^T \mid$ there exist Markov-chain random variables $S_1, S_2, \cdots, S_T$, such that $S_1 \Rightarrow S_2 \Rightarrow \cdots \Rightarrow S_T$ and $R_1 \le H(S_1),\ R_2 \le H(S_2|S_1),\ \cdots,\ R_T \le H(S_T|S_{T-1})\}$, which is exactly the capacity region of MLC with q levels nicely shown by Fu and Han Vinck in [4].

2) Outer Bound to Capacity Region: To derive an outer bound to the capacity region $\mathcal{A}_T$, we consider the rewriting code as defined in Definition 8, but with an additional property: The highest reachable levels $h_1, h_2, \cdots, h_N$ for the N VLC groups are known in advance. Thus the encoding and decoding functions can use that information. Let $\mathcal{A}_T^*$ denote its capacity region. Clearly, $\mathcal{A}_T^* \supseteq \mathcal{A}_T$, so it is an outer bound to $\mathcal{A}_T$.

Theorem 10. *Define* $\mathcal{G}_T = \{(R_1, R_2, \cdots, R_T) \in \mathbb{R}^T \mid$ *for* $i = 1, 2, \cdots, q-1$, *there exist* $(r_{1,i}, r_{2,i}, \cdots, r_{T,i}) \in \mathbb{R}^T$ *and Markov-chain random variables* $S_{1,i}, S_{2,i}, \cdots, S_{T,i}$ *taking values in* $\{0, 1, \cdots, i\}$, *such that*

$$\begin{aligned} & S_{1,i} \Rightarrow S_{2,i} \Rightarrow \cdots \Rightarrow S_{T,i}, \\ & r_{1,i} \le H(S_{1,i}), r_{2,i} \le H(S_{2,i}|S_{1,i}), \\ & \quad \cdots\cdots, \qquad r_{T,i} \le H(S_{T,i}|S_{T-1,i}), \end{aligned}$$

and for $j = 1, 2, \cdots, T$,

$$R_j = \sum_{k=1}^{q-1} p_1 p_2 \cdots p_k (1 - p_{k+1}) r_{j,k}. \quad \}$$

Let $\mathcal{C}_T$ *be the closed set generated by* $\mathcal{G}_T$. *We have* $\mathcal{A}_T^* = \mathcal{C}_T$.

Let $M_T \triangleq \max\{\sum_{t=1}^{T} R_t | (R_1, R_2, \cdots, R_T) \in \mathcal{A}_T\}$ denote the maximum total rate of all rewriting codes for VLC. It has been shown by Fu and Han Vinck in [4] that for WOM of $i+1$ levels, the maximum total rate over T writes is $\log_2 \binom{T+i}{i}$. By Theorem 10, we get $M_T \le \max\{\sum_{t=1}^{T} R_t | (R_1, R_2, \cdots, R_T) \in \mathcal{A}_T^*\} = \sum_{k=1}^{q-1} p_1 p_2 \cdots p_k (1 - p_{k+1}) \log_2 \binom{T+k}{k}$.

ACKNOWLEDGMENT

This work was supported in part by the NSF CAREER Award CCF-0747415, the NSF grant ECCS-0802107, and by an NSF-NRI award.

REFERENCES

[1] J. E. Brewer and M. Gill (*Ed.*), *Nonvolatile memory technologies with emphasis on flash*, John Wiley & Sons, Inc., Hoboken, New Jersey, 2008.

[2] T. M. Cover, "Enumerative source coding," *IEEE Transactions on Information Theory*, vol. IT-19, no. 1, pp. 73–77, Jan. 1973.

[3] I. Csiszar and J. Korner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, New York: Academic, 1981.

[4] F. 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. Inform. Theory*, vol. 45, no. 1, pp. 308-313, 1999.

[5] A. Jiang, V. Bohossian and J. Bruck, "Floating codes for joint information storage in write asymmetric memories," *Proc. IEEE Int. Symposium on Information Theory (ISIT)*, Nice, France, Jun. 2007, pp. 1166–1170.

[6] A. Jiang, H. Zhou and J. Bruck, "Variable-level cells for non-volatile memories," Caltech Technical Report, 2011. Online: *http* : *//www.paradise.caltech.edu/etr.html*.

[7] L. A. Lastras-Montano, M. Franceschini , T. Mittelholzer, J. Karidis and M. Wegman, "On the lifetime of multilevel memories," *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Seoul, Korea, 2009, pp. 1224-1228.

[8] H. T. Lue *et al.*, "Study of incremental step pulse programming (ISPP) and STI edge effect of BE-SONOS NAND flash," *Proc. IEEE Int. Symp. on Reliability Physics*, vol. 30, no. 11, pp. 693-694, May 2008.