Endurance-Limited Memories: Capacity and Codes

Yeow Meng Chee, *Senior Member, IEEE*, Michal Horovitz, *Member, IEEE*, Alexander Vardy, *Fellow, IEEE*, Van Khu Vu, and Eitan Yaakobi, *Senior Member, IEEE*

***Abstract*— *Resistive memories*, such as *phase change memories* and *resistive random access memories* have attracted significant attention in recent years due to their better scalability, speed, rewritability, and yet non-volatility. However, their *limited endurance* is still a major drawback that has to be improved before they can be widely adapted in large-scale systems. In this work, in order to reduce the wear out of the cells, we propose a new coding scheme, called *endurance-limited memories* (*ELM*) codes, that increases the endurance of these memories by limiting the number of cell programming operations. Namely, an ℓ-*change* t-write ELM code is a coding scheme that allows to write t messages into some n binary cells while guaranteeing that each cell is programmed at most ℓ times. In case $\ell = 1$, these codes coincide with the well-studied *write-once memory* (*WOM*) codes. We study some models of these codes which depend upon whether the encoder knows on each write the number of times each cell was programmed, knows only the memory state, or even does not know anything. For the decoder, we consider these similar three cases. We fully characterize the capacity regions and the maximum sum-rates of three models where the encoder knows on each write the number of times each cell was programmed. In particular, it is shown that in these models the maximum sum-rate is $\log \sum_{i=0}^{\ell} \binom{t}{i}$. We also study and expose the capacity regions of the models where the decoder is informed with the number of times each cell was programmed. Finally we present the most practical model where the encoder read the memory before encoding new data and the decoder has no information about the previous states of the memory.**

***Index Terms*— Endurance limited memories (ELM), rewriting codes, resistive memories.**

I. Introduction

EMERGING resistive memory technologies, such as *resistive random access memories* (*ReRAM*) and *phase-change memories* (*PCM*), have the potential to be the future's universal memories. They combine several important attributes starting from the speed of SRAM, the density of DRAM, and the non-volatility of flash memories. However, they fall short in their *write endurance*, which significantly increases their bit error rate (BER). Hence, solving the limited endurance of these memories is crucial before they can be widely adapted in large-scale systems [9], [18], [28].

Resistive memories are non-volatile memories which are composed of cells. The information is stored in the cells by changing their resistance. They combine the following two properties of DRAM and flash memories. Similarly to flash memories and unlike DRAM they are non-volatile memories and thus they don't require refresh operations. Furthermore, like DRAM and unlike flash memories they are rewritable without an erase operation. The main challenge that has remained to be solved in order to make these memories a legitimate candidate as a universal memory is their limited write endurance, which is the goal of this paper.

Endurance is defined as the number of set/reset cycles to switch the state of cells in ReRAM while it is still reliable. Owing to its importance, there are many researches which test and characterize the endurance of ReRAM in order to show a strong dependence on material of cells, cell size, [16], [26], and program operation [22]. To improve the endurance of ReRAM, recent research have focused on the structure and material of devices [16], [25] and programming schemes [6], [22]. In this work, we present a scheme to use rewriting code to improve the endurance lifetime of ReRAM.

Previous works have offered different solutions to combat the write endurance of resistive memories. In [12], the authors proposed to use locally repairable codes (LRC) in order to construct codes with small rewriting locality in order to mitigate both the problems of endurance and power consumption. In [27], the authors proposed mellow writes, a technique which is targeted to reduce the wear-out of the writes rather than reducing the number of writes. Lastly, several other works proposed coding schemes which correct stuck-at cells; see e.g. [14], [19], [24].

In order to combat the limited write endurance in resistive memories, this paper proposes to study the following new family of codes, called *endurance-limited memory* (*ELM*) codes. Assume there are n binary cells and t messages that are required to be stored in these cells sequentially. Assume also that each cell can be programmed at most $\ell \geqslant 1$ times. Then, we seek to find the set of achievable rates, i.e., the capacity region, and design code constructions for this model. Note that for $\ell = 1$, we get the classical problem of write-

Manuscript received November 2, 2020; revised September 15, 2021; accepted November 17, 2021. Date of publication December 6, 2021; date of current version February 17, 2022. An earlier version of this paper was presented in part at the 2018 International Symposium on Information Theory and Its Applications [4] [DOI: 10.23919/ISITA.2018.8664328] and in part at the 2019 IEEE Information Theory Workshop [5]. *(Corresponding author: Van Khu Vu.)*

Yeow Meng Chee and Van Khu Vu are with the Department of Industrial Systems Engineering and Management, National University of Singapore, Singapore 119077 (e-mail: ymchee@nus.edu.sg; isevvk@nus.edu.sg).

Michal Horovitz is with the Department of Computer Science, Tel-Hai College, Galilee Research Institute—Migal, Kiryat Shmona, Upper Galilee 11016, Israel (e-mail: horovitzmic@telhai.ac.il).

Alexander Vardy is with the Department of Electrical and Computer Engineering and the Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: avardy@ucsd.edu).

Eitan Yaakobi is with the Computer Science Department, Technion—Israel Institute of Technology, Haifa 3200003, Israel (e-mail: yaakobi@cs.technion.ac.il).

Communicated by C. Hollanti, Associate Editor for Coding Theory.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIT.2021.3132995.

Digital Object Identifier 10.1109/TIT.2021.3132995

0018-9448 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

once memory (WOM) codes [2], [10], [11], [17], [21], [23]. Besides that, if $t \leqslant \ell$, the coding scheme is trivial. Hence, in this work, we only focus on the cases where $t > \ell > 1$. Note that a trivial lower bound on the maximum sum-rate is $\min\{t, \ell\}$, which achieved by writing with rate 1 in the first $\min\{t, \ell\}$ writes and with rate 0 in the remaining writes.

Let us consider first the case where $\ell = 2$ and $t = 3$. A naive solution is to use a two-write WOM code for the first two writes and then write n more bits on the third write. The maximum sum-rate using this solution will be $\log(3) + 1 = \log(6)$, while, as will be shown in the paper, the maximum sum-rate in this case is $\log(7)$. The intuition behind this is as follows. Let p_1 be the probability to program a cell on the first write, so we assume that $p_1 n$ cells are programmed. Then, on the second and third writes we have a two-write WOM code problem for the $p_1 n$ programmed cells, and for the $(1 - p_1)n$ non-programmed cells, we can write twice on them so no coding is needed. The maximum sum-rate is achieved for $p_1 = 3/7$. However, it is still a challenging task to design specific code constructions that can approach sum-rate of $\log(7)$.

There are several models of ELM codes which can be studied. These models are distinguishable by the information that is available to the encoder and the decoder. In particular, for the encoder, we consider three cases which depend upon whether the encoder knows the number of times each cell was programmed, *encoder informed all* (*EIA*), only the current state of the cell, *encoder informed partially* (*EIP*), or no information about the cells state, *encoder uninformed* (*EU*). The decoder will also have three cases, corresponding to the same information that is available to the encoder. Thus, by considering all combinations of the above three cases for the encoder and the decoder, it is possible to define and study nine models, $EX : DY$, where $X, Y \in \{IA, IP, U\}$.

The rest of this paper is organized as follows. In Section II, we formally define the models studied in this paper and discuss some basic observations. In Section III, we study the capacity regions and the maximum sum-rates of the EIA models, and also present capacity achieving codes. We prove that the capacity region of all EIA models, are the same, for both ϵ-error and zero-error cases. In the next two sections, we discuss the EIP:DIA model. In Section IV, we study the capacity region of this model for the ϵ-error case, and in Section V, we compare between this model and the EIA models. Then, we discuss the EU:DIA and the EIP:DU models in Sections VI and VII, respectively. Finally, we conclude our results and discuss a future work in Section VIII.

II. Definitions and Preliminaries

In this section, we formally define the nine models of ELM codes, and we state some simple propositions. Assume that each cell can be programmed at most ℓ times, so if the encoder attempts to program a cell more than ℓ times then it will not change its value. For the EIA models, we assume that the encoder will not try to program a cell that has already been programmed ℓ times before the current write. We see this as an extension of the WOM model for $\ell = 1$. These models will be defined both for the zero-error and the ϵ-error cases.

For a positive integer a, the set $\{0, \ldots, a-1\}$ is defined by $[a]$. Throughout this paper, we assume that the number of cells is n. We use the vector notation $\boldsymbol{c} \in [2]^n$ to represent the *cell-state vector* of the n memory cells, and the vector $\boldsymbol{v} \in [\ell+1]^n$, which will be called the *cell-program-count vector*, to represent the number of times each cell was programmed. Note that the state of a cell is the parity of the number of times it was programmed. Thus, if the encoder (or the decoder) knows the cell-program-count vector $\boldsymbol{v}$, in particular it knows the cell-state vector $\boldsymbol{c}$ as well. For a vector $\boldsymbol{v} \in [\ell+1]^n$, we denote by $\langle \boldsymbol{v} \rangle_2$ the length-n binary vector which satisfies $\langle \boldsymbol{v} \rangle_{2,k} = \boldsymbol{v}_k \pmod 2$ for all $k \in [n]$, and we say that $\langle \boldsymbol{v} \rangle_2$ equals to $\boldsymbol{v}$ modulo 2. The complement of a binary vector $\boldsymbol{c}$ is denoted by $\overline{\boldsymbol{c}}$. The all ones, zeros vector will be denoted by $\mathbf{1}, \mathbf{0}$, respectively. For two length-n vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, $\boldsymbol{a}+\boldsymbol{b}$ is the vector obtained by pointwise addition. If $\boldsymbol{a}$ and $\boldsymbol{b}$ are binary vectors $\boldsymbol{a} \oplus \boldsymbol{b}$ is the vector obtained by pointwise addition modulo 2.

For a cell-program-count vector $\boldsymbol{v} \in [\ell+1]^n$ and a new cell-state vector $\boldsymbol{c} \in [2]^n$ to be programmed to the cells, we define by $N(\boldsymbol{v}, \boldsymbol{c}) \in [\ell+1]^n$ and $f(\boldsymbol{v}, \boldsymbol{c}) \in [2]^n$ the result of programming the new cell-state vector $\boldsymbol{c}$. That is, $N(\boldsymbol{v}, \boldsymbol{c})$ is the new cell-program-count vector after programming $\boldsymbol{c}$, $f(\boldsymbol{v}, \boldsymbol{c})$ is the new cell-state vector, and they are formally defined as follows. $N(\boldsymbol{v}, \boldsymbol{c})_k = \boldsymbol{v}_k$ if $\boldsymbol{c}_k = \boldsymbol{v}_k \pmod 2$, and otherwise $N(\boldsymbol{v}, \boldsymbol{c})_k = \min\{\ell, \boldsymbol{v}_k + 1\}$, where the index k for a vector means its k-th element. Similarly, $f(\boldsymbol{v}, \boldsymbol{c})_k = \boldsymbol{c}_k$ if $\boldsymbol{v}_k < \ell$, and otherwise $f(\boldsymbol{v}, \boldsymbol{c})_k = \boldsymbol{v}_k \pmod 2$. Note that $\langle N(\boldsymbol{v}, \boldsymbol{c}) \rangle_2 = f(\boldsymbol{v}, \boldsymbol{c})$, i.e., $f(\boldsymbol{v}, \boldsymbol{c})$ equals to $N(\boldsymbol{v}, \boldsymbol{c})$ modulo 2. We are ready now to define all models studied in this paper.

Definition 1: An $[n, t, \ell; M_1, \ldots, M_t]^{EX:DY, \boldsymbol{p}_e}$ ℓ-change t-write **Endurance-Limited Memory** (**ELM**) code with error-probability vector $\boldsymbol{p}_e = (p_{e_1}, \ldots, p_{e_t})$, where $X, Y \in \{IA, IP, U\}$, is a coding scheme comprising of n binary cells and is defined by t encoding and decoding maps $(\mathcal{E}_j, \mathcal{D}_j)$ for $1 \leqslant j \leqslant t$. For the map $\mathcal{E}_j$, $Im(\mathcal{E}_j)$ is its image, where by definition $Im(\mathcal{E}_0) = \{(0, \ldots, 0)\}$.

Furthermore, for $j \in [t+1]$, let N_j and $Im^*(\mathcal{E}_j)$ be the sets of all state-program-count vectors, cell-state vectors which can be obtained after the first j writes, respectively. Formally, for $1 \leqslant j$, $N_j = \{N(\boldsymbol{v}, \boldsymbol{c}) : \boldsymbol{c} \in Im(\mathcal{E}_j), \boldsymbol{v} \in N_{j-1}\}$, where $N_0 = \{(0, \ldots, 0)\}$, and $Im^*(\mathcal{E}_j) = \{\langle \boldsymbol{v} \rangle_2 : \boldsymbol{v} \in N_j\}$. Note that for the EIA models $Im^*(\mathcal{E}_j) = Im(\mathcal{E}_j)$. The domain and the range of the encoding maps are defined as follows:

(1) for the EIA models, $\mathcal{E}_j : [M_j] \times N_{j-1} \to [2]^n$, such that for all $(m, \boldsymbol{v}) \in [M_j] \times N_{j-1}$ it holds that $\boldsymbol{v} + (\langle \boldsymbol{v} \rangle_2 \oplus \mathcal{E}_j(m, \boldsymbol{v})) \in [\ell+1]^n$.
(2) for the EIP models, $\mathcal{E}_j : [M_j] \times Im^*(\mathcal{E}_{j-1}) \to [2]^n$.
(3) for the EU models, $\mathcal{E}_j : [M_j] \to [2]^n$.

For a message m we denote by $I_m(x)$ the indicator function, where $I_m(x) = 0$ if $m = x$, and otherwise $I_m(x) = 1$. Additionally, for a message $m \in [M_j]$, $Pr(m)$ is the probability of programming a message m on the j-th write, and for $\boldsymbol{v} \in N_{j-1}$, $Pr(\boldsymbol{v})$ is the probability to have cell-program-count vector $\boldsymbol{v}$ before the j-th write. The nine models are defined as follows. For all $1 \leqslant j \leqslant t$,

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

(1) if $(X,Y)=(IA,IA)$ then

$$\mathcal{D}_j:\{(\mathcal{E}_j(m,\boldsymbol{v}),\boldsymbol{v}):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(\mathcal{E}_j(m,\boldsymbol{v}),\boldsymbol{v})\right)\leqslant p_{e_j},$$

(2) if $(X,Y)=(IA,IP)$ then

$$\mathcal{D}_j:\{(\mathcal{E}_j(m,\boldsymbol{v}),\langle\boldsymbol{v}\rangle_2):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(\mathcal{E}_j(m,\boldsymbol{v}),\langle\boldsymbol{v}\rangle_2)\right)\leqslant p_{e_j},$$

(3) if $(X,Y)=(IA,U)$ then

$$\mathcal{D}_j:Im(\mathcal{E}_j)\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(\mathcal{E}_j(m,\boldsymbol{v}))\right)\leqslant p_{e_j},$$

(4) if $(X,Y)=(IP,IA)$ then

$$\mathcal{D}_j:\{(f(\boldsymbol{v},\mathcal{E}_j(m,\langle\boldsymbol{v}\rangle_2)),\boldsymbol{v}):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m,\langle\boldsymbol{v}\rangle_2)),\boldsymbol{v})\right)\leqslant p_{e_j},$$

(5) if $(X,Y)=(IP,IP)$ then

$$\mathcal{D}_j:\{(f(\boldsymbol{v},\mathcal{E}_j(m,\langle\boldsymbol{v}\rangle_2)),\langle\boldsymbol{v}\rangle_2):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m,\langle\boldsymbol{v}\rangle_2)),\langle\boldsymbol{v}\rangle_2)\right)\leqslant p_{e_j},$$

(6) if $(X,Y)=(IP,U)$ then

$$\mathcal{D}_j:Im^*(\mathcal{E}_j)\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m,\langle\boldsymbol{v}\rangle_2)))\right)\leqslant p_{e_j},$$

(7) if $(X,Y)=(U,IA)$ then

$$\mathcal{D}_j:\{(f(\boldsymbol{v},\mathcal{E}_j(m)),\boldsymbol{v}):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m))\boldsymbol{v})\right)\leqslant p_{e_j},$$

(8) if $(X,Y)=(U,IP)$ then

$$\mathcal{D}_j:\{(f(\boldsymbol{v},\mathcal{E}_j(m)),\langle\boldsymbol{v}\rangle_2):m\in[M_j],\boldsymbol{v}\in N_{j-1}\}\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m))\langle\boldsymbol{v}\rangle_2)\right)\leqslant p_{e_j},$$

(9) if $(X,Y)=(U,U)$ then

$$\mathcal{D}_j:Im^*(\mathcal{E}_j)\to[M_j],$$

and

$$\sum_{(m,\boldsymbol{v})\in[M_j]\times N_{j-1}} Pr(m)Pr(\boldsymbol{v})\cdot I_m\left(\mathcal{D}_j(f(\boldsymbol{v},\mathcal{E}_j(m)))\right)\leqslant p_{e_j}.$$

If $p_{e_j}=0$ for all $1\leqslant j\leqslant t$, then the code is called a zero-error ELM code and is denoted by $[n,t,\ell;M_1,\ldots,M_t]^{EX:DY,z}$.

The *rate* on the j-th write of an $[n,t,\ell;M_1,\ldots,M_t]^{EX:DY,\boldsymbol{p}_e}$ ELM code, $X,Y\in\{IA,IP,U\}$, is defined as $R_j=\frac{\log M_j}{n}$, and the *sum-rate* is the sum of the individual rates on all writes, $R_{sum}=\sum_{j=1}^{t}R_j$. A rate tuple $\mathbf{R}=(R_1,\ldots,R_t)$ is called *ϵ-error achievable* in model $EX:DY$, if for all $\epsilon>0$ there exists an $[n,t,\ell;M_1,\ldots,M_t]^{EX:DY,\boldsymbol{p}_e}$ ELM code with error-probability vector $\boldsymbol{p}_e=(p_{e_1},\ldots,p_{e_t})\leqslant(\epsilon,\ldots,\epsilon)$, such that $\frac{\log M_j}{n}\geqslant R_j-\epsilon$. The rate tuple $\mathbf{R}$ will be called *zero-error achievable* if for all $1\leqslant j\leqslant t$, $p_{e_j}=0$. The *ϵ-error capacity region* of the EX:DY model is the set of all ϵ-error achievable rate tuples, that is,

$$\mathcal{C}_{t,\ell}^{EX:DY,\epsilon}=\{(R_1,\ldots,R_t)|(R_1,\ldots,R_t)\text{ is }\epsilon\text{-error achievable}\},$$

and the *ϵ-error maximum sum-rate* will be denoted by $\mathcal{R}_{t,\ell}^{EX:DY,\epsilon}$. The *zero-error capacity region* $\mathcal{C}_{t,\ell}^{EX:DY,z}$ and the *zero-error maximum sum-rate* $\mathcal{R}_{t,\ell}^{EX:DY,z}$ are defined similarly. We say that $\mathbf{R}\leqslant\mathbf{R}'$ for $\mathbf{R}=(R_1,\ldots,R_t)$ and $\mathbf{R}'=(R'_1,\ldots,R'_t)$ if $R_j\leqslant R'_j$ for all $1\leqslant j\leqslant t$, and $\mathbf{R}<\mathbf{R}'$ if $\mathbf{R}\leqslant\mathbf{R}'$ and $\mathbf{R}\neq\mathbf{R}'$.

According to these definitions it is easy to verify the following relations. For $g\in\{z,\epsilon\}$ and $X,Y\in\{IA,IP,U\}$ it holds that

$$\mathcal{C}_{t,\ell}^{EU:DY,g}\subseteq\mathcal{C}_{t,\ell}^{EIP:DY,g}\subseteq\mathcal{C}_{t,\ell}^{EIA:DY,g},$$
$$\mathcal{C}_{t,\ell}^{EX:DU,g}\subseteq\mathcal{C}_{t,\ell}^{EX:DIP,g}\subseteq\mathcal{C}_{t,\ell}^{EX:DIA,g},\text{ and}$$
$$\mathcal{C}_{t,\ell}^{EX:DY,z}\subseteq\mathcal{C}_{t,\ell}^{EX:DY,\epsilon}.$$

Similar connections hold for the maximum sum-rates.

Note that if $\ell\geqslant t$ then all problems are trivial since it is possible to program all cells on each write, so the capacity region in all models is $[0,1]^t$ and the maximum sum-rate is t. For $\ell=1$, we get the classical and well-studied WOM codes [2], [10], [11], [17], [21], [23]. In this case, we also notice that the IA and IP models are the same for both the encoder and the decoder. The capacity region and the maximum sum-rate in most of these cases are known; see e.g. [10], [11], [17], [23]. In the rest of this paper, and unless stated otherwise, we assume that $1\leqslant\ell<t$.

A. Related Work

The EIA models of ELM codes studied in this paper are strongly related to non-binary WOM codes and their modified versions studied in [3], [8], [11], [13]. In these EIA models, we can treat every cell as an $(\ell+1)$-ary cell, where it is only possible to increase its level by one on each write, while its maximum level is ℓ. In non-binary WOM codes,

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

each cell has q levels and its level can not be decreased [8], [11]. In a write ℓ-step-up memory [3], a special version of the non-binary WOM code, each cell has q levels and each time we write a cell, its level can be increased by at most some value ℓ. Recently, Kobayashi *et al.* [13] also studied a modified version of non-binary WOM codes, called write constrained memories, where there is a cost on each state transition. Yet, these codes are not identical and we can not apply previous results to solve the models of ELM codes. In fact, our results on EIA models are useful to obtain an explicit formula of the capacity region of write ℓ-step-up memory codes. We also note that some models of ELM codes, such as the EIP:DU model, are much different from previous models and are difficult to solve.

Moreover, our proposed ELM coding scheme is also related to the cooling code which is used to control the peak temperature of an interconnect [1], [2]. In [1], cooling codes are proposed to avoid all hottest wires and in [2], cooling codes are shown to be equivalent to two-write binary WOM codes. Later in this work, we will use cooling codes as two-write binary WOM codes in order to construct our ELM codes. Now, we discuss the ability of our ELM codes to control the peak temperature of an interconnection. In fact, using our coding scheme in this work, we can control the maximal number of switches in each wire. We note that the temperature of each wire is closely related to the number of switches in each wire. And thus, we can control the peak temperature of each wire.

Recently, the EIA:DIA model of ELM is shown to be useful for two-dimensional (2D) weight-constrained code scheme [15]. In a 2D weight-constrained code, each codeword is an array of size $m \times n$ where the number of 1 symbols in each row and each column is limited by pn and qm, respectively. The encoder and decoder know the values of all the mn bits. We can view this 2D weight-constrained code as we write m messages in ReRAM where each cell can be switched at most qm times and both encoder and decoder know all previous messages.

B. Our Contribution

In this work, we propose a novel scheme of rewriting code to improve the endurance lifetime of ReRAM, called endurance limited memories (ELM) code. In ℓ-change t-write ELM codes, each cell can be programmed at most ℓ times during the writing of t messages. In the case that ℓ is much smaller than t, we can significantly improve the lifetime of the memories. Depending upon whether the encoder (E) and decoder (D) know the number of times each cell is programmed (IA), the current state of each cell (IP), or have no information on the state of each cell (U), we present and investigate all nine models $EX : DY$ where $X, Y \in \{IA, IP, U\}$. We note that the most practical model to increase the endurance of ReRAM is the EIP:DU model. However, for the theoretical interest, we study all nine models in this paper. Furthermore, although the EIA models are not suitable for improving the endurance of ReRAM, they have several applications, including write ℓ-step-up memories and two-dimensional weight-constrained codes. We expect that we can see other applications of all nine models of ELM codes in near future.

In Section III, all three EIA models are investigated in both cases, ϵ-error and zero-error. We first provide the capacity region and the maximum sum-rate of these models. The techniques are used to achieve the results on the capacity regions in Theorems 2 and 3 are similar to those used in [11] for WOM codes. To achieve the explicit formula of the maximum sum-rate in Theorem 4, we need a new simple technique. We then present several constructions of ELM codes for the EIA models. To construct these codes, we need some new ideas even though we also use several special families of WOM codes as components of our ELM codes. In Section IV, we use a known technique in information theory to obtain the capacity region of the EIP:DIA model. The capacity region of all EIA models is compared to the EIP:DIA model in Section V. Using the same technique carefully, we can also find the capacity region of the EU:DIA model in Section VI. Finally, in Section VII, we study the most practical model for ReRAM, the EIP:DU model. Although several good bounds on the maximum sum-rate are presented, these bounds are not tight and finding an exact formula of the maximum sum-rate of the EIP:DU model is still an open problem. To achieve some good constructive lower bounds, we provide several constructions of EIP:DU ELM codes for the zero-error case. These results are novel and we need different original methods to achieve them.

III. The EIA Models

In this section, we explore the capacity region and the maximum sum-rate of the EIA models for both the ϵ-error and the zero-error cases. We also propose capacity achieving codes for these cases.

For each $j \in [t+1]$, we let $\boldsymbol{c}_j$ denote the binary length-n vector which represents the cell-state vector after the j-th write, where $\boldsymbol{c}_0 = \mathbf{0}$. Recall that in the EIA models, on the j-th write the encoder knows the number of times each cell was programmed before the current write. That is, the encoder receives as an input a length-n cell-program-count vector $N(\boldsymbol{c}_{j-1}) \in [\ell+1]^n$ that represents the number of times each cell was programmed so far. Next, for all t and ℓ, we define the region $\mathcal{C}_{t,\ell}$ and in Theorem 2, we prove that this is the capacity region of all the EIA models.

For $1 \leqslant j \leqslant t$ and $i \in [\ell+1]$, $i \leqslant j$, let $p_{j,i} \in [0, 0.5]$, be the probability to program a cell on the j-th write, given that this cell has been already programmed i times. We define $p_{j,j} = p_{j,\ell} = 0$ for $1 \leqslant j \leqslant t$, and let $Q_{j,i}$ be the probability that a cell has been programmed exactly i times on the first j writes. Formally, $Q_{j,i}$ is defined recursively by using $p_{j,i}$ and $p_{j,i-1}$ as follows:

$$Q_{j,i} = \begin{cases} Q_{j-1,i}(1-p_{j,i}) + Q_{j-1,i-1}p_{j,i-1}, & \text{if } i > 0, \\ Q_{j-1,i}(1-p_{j,i}), & \text{if } i = 0, \end{cases} \tag{1}$$

where for $j = 0$ we set $Q_{0,0} = 1$ otherwise $Q_{0,i} = 0$. The rates region $\mathcal{C}_{t,\ell}$ is defined as follows:

$$\mathcal{C}_{t,\ell} = \Big\{ (R_1, \ldots, R_t) | \forall 1 \leqslant j \leqslant t : R_j \leqslant \sum_{i=0}^{\min\{\ell,j\}-1} Q_{j-1,i} h(p_{j,i}), \\ \forall i \in [\ell] : p_{j,i} \in [0, 0.5], \text{ and } Q_{j,i} \text{ is defined in (1)} \Big\}, \tag{2}$$

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

where in this paper both $h(x)$ and $H(X)$ represent the binary entropy function for $0 \leqslant x \leqslant 1$ and a random variable X, respectively.

Note that for $\ell = 1$, it is possible to verify that we get the capacity region of WOM [10], [17], [23]. It is also readily verified that the maximum sum-rate is achieved with $p_{j,i} = 0.5$ for all $t - j + 1 \geqslant \ell - i$, since $t - j + 1$ is the number of the remaining writes, and $\ell - i$ is the number of times the cell can be programmed. Thus, if $t - j + 1 \geqslant \ell - i$ then we can program the cell with probability 0.5 to obtain the maximum rate. The next theorem proves that for $2 \leqslant \ell \leqslant t - 1$, $\mathcal{C}_{t,\ell}$ is the capacity region of the ℓ-change t-write ELM in all EIA models, and thus we denote this capacity by $\mathcal{C}_{t,\ell}^{EIA}$, and the maximum sum-rate by $\mathcal{R}_{t,\ell}^{EIA}$.

Theorem 2: The rates region $\mathcal{C}_{t,\ell}$ is the capacity region of the ℓ-change t-write ELM in all EIA models for both ϵ-error and zero-error cases. That is, for all $g \in \{z, \epsilon\}$ and $Y \in \{IA, IP, U\}$, $\mathcal{C}_{t,\ell}^{EIA:DY,g} = \mathcal{C}_{t,\ell}$.

Proof: Recall that by the definitions of the models

$$\mathcal{C}_{t,\ell}^{EIA:DU,z} \subseteq \mathcal{C}_{t,\ell}^{EIA:DIP,z} \subseteq \mathcal{C}_{t,\ell}^{EIA:DIA,z} \subseteq \mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon}, \text{and}$$
$$\mathcal{C}_{t,\ell}^{EIA:DU,z} \subseteq \mathcal{C}_{t,\ell}^{EIA:DU,\epsilon} \subseteq \mathcal{C}_{t,\ell}^{EIA:DIP,\epsilon} \subseteq \mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon}.$$

The rest of the proof consists of two parts. The first part, called the direct part, proves that $\mathcal{C}_{t,\ell} \subseteq \mathcal{C}_{t,\ell}^{EIA:DU,z}$, and in the second, called the converse part, we prove that $\mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon} \subseteq \mathcal{C}_{t,\ell}$. The direct part is proved in Subsection III-A for the zero-error case of the EIA:DU model, while the converse part is proved in Subsection III-B for the ϵ-error case of the EIA:DIA model. ∎

Next, we seek to present the capacity region of the EIA models in a recursive form. While we see this representation of the capacity region more intuitive, it will also help us in finding the maximum sum-rate of this model. For all $t \geqslant 1$ and $\ell \geqslant 1$, let $\widehat{\mathcal{C}}_{t,\ell}$ be the following region which is defined recursively as follows. For $t > \ell \geqslant 1$

$$\begin{aligned}\widehat{\mathcal{C}}_{t,\ell} = \Big\{ & (R_1, \ldots, R_t) | R_1 \leqslant h(p), p \in [0, 0.5], \\ & \text{for } 2 \leqslant j \leqslant t, R_j \leqslant pR'_j + (1-p)R''_j, \\ & (R'_2, \ldots, R'_t) \in \widehat{\mathcal{C}}_{t-1,\ell-1} \text{ and } (R''_2, \ldots, R''_t) \in \widehat{\mathcal{C}}_{t-1,\ell} \Big\},\end{aligned} \tag{3}$$

where for all $\ell \geqslant t \geqslant 1$ we set $\widehat{\mathcal{C}}_{t,\ell} = [0,1]^t$ and $\widehat{\mathcal{C}}_{t,0} = \{\mathbf{0}\}$.

Theorem 3: For all t and ℓ, $\widehat{\mathcal{C}}_{t,\ell} = \mathcal{C}_{t,\ell}$.

Proof: For the first direction, we prove by induction on t that for all $\ell \geqslant 1$, if $\mathbf{R} = (R_1, \ldots, R_t) = \widehat{\mathcal{C}}_{t,\ell}$ then $\mathbf{R} \in \mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon}$. Since $\mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon} = \mathcal{C}_{t,\ell}$, we conclude that $\widehat{\mathcal{C}}_{t,\ell} \subseteq \mathcal{C}_{t,\ell}$.

The base of the induction is $t \leqslant \ell$ for all $\ell \geqslant 1$. These cases are readily verified. For the induction step, let $\mathbf{R} = (R_1, R_2, \ldots, R_t) \in \widehat{\mathcal{C}}_{t,\ell}$, $1 \leqslant \ell < t$, such that $R_1 = h(p)$ for $p \in [0, 0.5]$ and for $2 \leqslant j \leqslant t$ $R_j = pR'_j + (1-p)R''_j$ where $(R'_2, R'_3, \ldots, R'_t) \in \widehat{\mathcal{C}}_{t-1,\ell-1}$ and $(R''_2, R''_3, \ldots, R''_t) \in \widehat{\mathcal{C}}_{t-1,\ell}$. By the induction hypothesis, $(R'_2, R'_3, \ldots, R'_t) \in \mathcal{C}_{t-1,\ell-1}^{EIA:DI}$ and $(R''_2, R''_3, \ldots, R''_t) \in \mathcal{C}_{t-1,\ell}^{EIA:DI}$. Thus, we have two codes: C_1 - an $(\ell-1)$-change $(t-1)$-write ELM code which achieves the rate tuple $(R'_2, R'_3, \ldots, R'_t)$ and C_2 - an ℓ-change $(t-1)$-write ELM code which achieves the rate tuple $(R''_2, R''_3, \ldots, R''_t)$. Then, we can design an ℓ-change t-write ELM code, such that on the first write the encoder programs a cell with probability p for $p \in [0, 0.5]$, and then on the next writes it applies C_1 for the cells that were programmed on the first write, and C_2 for the other cells. Thus, the rate tuple $\mathbf{R}$ is achieved.

The second direction, $\mathcal{C}_{t,\ell} \subseteq \widehat{\mathcal{C}}_{t,\ell}$, is proved by induction on t, that is, for each $t \geqslant 1$ we prove that $\mathcal{C}_{t,\ell} \subseteq \widehat{\mathcal{C}}_{t,\ell}$ for all $1 \leqslant \ell \leqslant t$. The base of the induction, $t = 1$ and $\ell = 1$, is trivial. The induction assumption is that for each $1 \leqslant \ell' \leqslant t-1$, $\mathcal{C}_{t-1,\ell'} \subseteq \widehat{\mathcal{C}}_{t-1,\ell'}$. For the induction step, let $\mathbf{R} = (R_1, R_2, \ldots, R_t) \in \mathcal{C}_{t,\ell}$ which is achieved by the probabilities $p_{j,i}$. Denote by $\mathbf{R}' = (R'_2, R'_3, \ldots, R'_t) \in \mathcal{C}_{t-1,\ell-1}$ the rate tuple which is attained by the probabilities $p'_{j,i} = p_{j+1,i+1}$, and by $\mathbf{R}'' = (R''_2, R''_3, \ldots, R''_t) \in \mathcal{C}_{t-1,\ell}$ the rate tuple which is attained by the probabilities $p''_{j,i} = p_{j+1,i}$. Recall that we define $\widehat{\mathcal{C}}_{t-1,t} = \widehat{\mathcal{C}}_{t-1,t-1}$, and $\widehat{\mathcal{C}}_{t-1,0} = \{\mathbf{0}\}$. It can be easily verified that for all j, $2 \leqslant j \leqslant t$, $R_j = p_{1,0}R'_j + (1-p_{1,0})R''_j$. By the induction hypothesis, $\mathbf{R}' \in \widehat{\mathcal{C}}_{t-1,\ell-1}$ and $\mathbf{R}'' \in \widehat{\mathcal{C}}_{t-1,\ell}$. Thus, by defining $p = p_{1,0}$ we get a recursive form for $\mathbf{R}$, and we can conclude that $\mathcal{C}_{t,\ell} \subseteq \widehat{\mathcal{C}}_{t,\ell}$. ∎

Next, using the result from Theorem 3, it is possible to find the maximum sum-rate of the EIA models, $\mathcal{R}_{t,\ell}^{EIA}$.

Theorem 4: For all t and ℓ,

$$\mathcal{R}_{t,\ell}^{EIA} = \log \sum_{i=0}^{\ell} \binom{t}{i},$$

and this value is achieved for

$$p_{1,0} = p = \frac{\sum_{i=0}^{\ell-1} \binom{t-1}{i}}{\sum_{i=0}^{\ell} \binom{t}{i}},$$

where $p_{1,0}$, p are defined in Equations (2), (3) in $\mathcal{C}_{t,\ell}$, $\widehat{\mathcal{C}}_{t,\ell}$, respectively. For example, if $\ell = 2$ the maximum sum-rate is achieved for $p_{1,0} = p = \frac{2t}{t^2+t+2}$.

Proof: First, we prove that $\mathcal{R}_{t,\ell}^{EIA} \leqslant \log \sum_{i=0}^{\ell} \binom{t}{i}$ by counting all the possible sequences of t messages. We describe each possible sequence as a table of t rows and n columns, where n is the number of cells. Note that different sequences will be mapped to different matrices. Recall, that every cell can be programmed at most ℓ times. Thus, the number of different possible matrices is $\left(\sum_{i=0}^{\ell} \binom{t}{i}\right)^n$, and the upper bound is proved.

Next we assure that this upper bound is indeed tight. We prove this result by using the recursive formula for the capacity $\widehat{\mathcal{C}}_{t,\ell}$ described in Equation (3). For $\ell = 1$, WℓM is the binary WOM, and this upper bound, $\log(t+1)$ is known to be tight, and achieved for $p = 1/(t+1)$ [8], [10], [23]. That is, the maximum sum-rate of one-change t-write WℓM is equal to $\log \sum_{i=0}^{\ell} \binom{t}{i} = \log(t+1)$. Let us denote $X_{t,\ell} = \sum_{i=0}^{\ell} \binom{t}{i}$ and $p = \frac{X_{t-1,\ell-1}}{X_{t,\ell}}$. Note that by the properties of the binomial coefficients, $X_{t,\ell} = X_{t-1,\ell-1} + X_{t-1,\ell}$. Therefore, $1 - p = \frac{X_{t-1,\ell}}{X_{t,\ell}}$. By using the recursive formula for the capacity $\widehat{\mathcal{C}}_{t,\ell}$ described in Equation (3), we are only left to prove that for

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

all $2 \leqslant \ell \leqslant t-1$,

$$\log X_{t,\ell} = h(p) + p \log X_{t-1,\ell-1} + (1-p) \log X_{t-1,\ell}.$$

This relation holds since

$$\begin{aligned} &h(p) + p \log X_{t-1,\ell-1} + (1-p) \log X_{t-1,\ell} \\ &= p \left(\log \left(\tfrac{X_{t,\ell}}{X_{t-1,\ell-1}} \right) + \log(X_{t-1,\ell-1}) \right) \\ &+ (1-p) \left(\log \left(\tfrac{X_{t,\ell}}{X_{t-1,\ell}} \right) + \log(X_{t-1,\ell}) \right) \\ &= p \log X_{t,\ell} + (1-p) \log(X_{t,\ell}) \\ &= \log(X_{t,\ell}). \end{aligned}$$

■

A. The EIA:DU Model - Constructions and Direct Part of Theorem 2

In this subsection, we study the EIA:DU model, that is, encoder informed all and decoder uninformed. Our main contribution is a construction of a capacity-achieving ℓ-change t-write EIA:DU-ELM code for the zero-error case, which assures that $\mathcal{C}_{t,\ell} \subseteq \mathcal{C}_{t,\ell}^{EIA:DU,z}$. That is, the direct part of Theorem 2 is proved.

Let us start with the first non-trivial case of $t=3$ and $\ell=2$. Thus, we want to prove that $\mathcal{C}_{3,2} \subseteq \mathcal{C}_{3,2}^{EIA:DU,z}$. Recall that,

$$\begin{aligned} \mathcal{C}_{3,2} = \Big\{ &(R_1, R_2, R_3) | R_1 \leqslant h(p_{1,0}), \\ &R_2 \leqslant 1 - p_{1,0} + p_{1,0} h(p_{2,1}), \\ &R_3 \leqslant 1 - p_{1,0} p_{2,1}, \text{and} \, p_{1,0}, p_{2,1} \in [0, 0.5] \Big\}. \end{aligned}$$

which is achieved by setting $p_{3,0} = p_{3,1} = p_{2,0} = 0.5$ in Equation (2). The next theorem states the existence of a construction of ELM codes for this case.

Theorem 5: For any $\epsilon > 0$ and $p_{1,0}, p_{2,0}, p_{2,1} \in [0, 0.5]$, there exists an explicit construction of a zero-error two-change three-write EIA:DU-ELM code satisfying $R_1 \geqslant h(p_{1,0}) - \epsilon$, $R_2 \geqslant (1-p_{1,0}) h(p_{2,0}) + p_{1,0} h(p_{2,1}) - \epsilon$, and $R_3 \geqslant (1 - p_{1,0} p_{2,1}) - \epsilon$.

Before presenting our construction for two-change three-write EIA:DU-ELM codes, we introduce the following family of WOM codes. We then use these WOM codes as component codes in our construction of EIA:DU-ELM codes. Note that the WOM codes we use for our construction are given for $n \to \infty$, and thus our constructions for ELM codes use such n.

Definition 6: An $[n, 2; M_1, M_2]_q^{EI:DU,z}$ two-write q-ary EI:DU WOM code for the zero-error case is a coding scheme comprising of n q-ary cells. It consists of two pairs of encoding and decoding maps $(\mathcal{E}_{q,1}, \mathcal{D}_{q,1})$ and $(\mathcal{E}_{q,2}, \mathcal{D}_{q,2})$ which are defined as follows:

(1) $\mathcal{E}_{q,1} : [M_1] \to [q]^n$ and $\mathcal{D}_{q,1} : Im(\mathcal{E}_{q,1}) \to [M_1]$ such that for all $m_1 \in [M_1]$, $\mathcal{D}_{q,1}(\mathcal{E}_{q,1}(m_1)) = m_1$.
(2) $\mathcal{E}_{q,2} : [M_2] \times Im(\mathcal{E}_{q,1}) \to [q]^n$ and $\mathcal{D}_{q,2} : Im(\mathcal{E}_{q,2}) \to [M_2]$ such that for all $(m_2, \boldsymbol{c}) \in [M_2] \times Im(\mathcal{E}_{q,1})$, $\mathcal{E}_{q,2}(m_2, \boldsymbol{c}) \geqslant \boldsymbol{c}$ and $\mathcal{D}_{q,2}(\mathcal{E}_{q,2}(m_2, \boldsymbol{c})) = m_2$.

We say that $\boldsymbol{p} = (p_0, p_1, \ldots, p_{m-1})$ is a *probability vector* if $\sum_{i=0}^{m-1} p_i = 1$ and $p_i \geqslant 0$ for all $i \in [m]$. We distinguish between an error-probability vector that is used in Definition 1, and a probability vector. An error-probability vector is a vector of error-probabilities, and not a probability vector, i.e., the sum of the elements of an error-probability vector does not need be 1. For two positive integers n, q and a probability vector $\boldsymbol{p} = (p_0, p_1, \ldots, p_{q-1})$, we denote by $\mathcal{B}(n, \boldsymbol{p})$ the set of all length-n q-ary vectors of constant composition $\boldsymbol{w} = (w_0, \ldots, w_{q-1})$, where $w_i = p_i n$ for $i \in [q]$ [1]. Let $p_{j,i \to k}$ be the probability that on the j-th write, a cell in state i is programmed to state k, $k \geqslant i$.

A family of two-write q-ary capacity-achieving EI:DU WOM codes was constructed recently by Shpilka [21]. Particularly, given $\epsilon > 0$ and probability vectors $\boldsymbol{p}_{1,0}$, $\boldsymbol{p}_{2,0}, \ldots, \boldsymbol{p}_{2,q-2}$, Shpilka [21] constructed a family of two-write q-ary EI:DU WOM codes that match these probability vectors on the first and second writes. We state this result formally.

Lemma 7: [21] For all $(j,i) \in \{(1,0), (2,0), (2,1), \ldots, (2, q-2)\}$, let $\boldsymbol{p}_{j,i} = (p_{j,i \to i}, p_{j,i \to i+1}, \ldots, p_{j,i \to q-1})$ be a probability vector. Then, for all $\epsilon > 0$ there exists an $[n, 2; M_1, M_2]_q^{EI:DU,z}$ two-write q-ary EI:DU WOM code satisfying:

- $Im(\mathcal{E}_{q,1}) \subseteq \mathcal{B}(n, \boldsymbol{p}_{1,0})$ and $R_1 = \frac{\log M_1}{n} \geqslant h(\boldsymbol{p}_{1,0}) - \epsilon$.
- For all $\boldsymbol{c}_1 \in Im(\mathcal{E}_{q,1})$, $m_2 \in [M_2]$, and $\boldsymbol{c}_2 = \mathcal{E}_{q,2}(m_2, \boldsymbol{c}_1)$, the following condition holds. For $i \in [q]$, let $\boldsymbol{c}_2^i$ be a length-$w_{1,i}$, $w_{1,i} = n p_{1,0 \to i}$, substring of $\boldsymbol{c}_2$ at all locations k with value i before the second write, that is, $\boldsymbol{c}_{1,k} = i$. Then, $\boldsymbol{c}_2^i \in \mathcal{B}(w_{1,i}, \boldsymbol{p}_{2,i})$. Furthermore, $R_2 = \frac{\log M_2}{n} \geqslant \sum_{i=0}^{q-2} p_{1,0 \to i} h(\boldsymbol{p}_{2,i}) - \epsilon$.

We refer to the family of WOM codes from Lemma 7 as an $[n, 2; M_1, M_2]_q^{EI:DU}(\epsilon, \boldsymbol{p}_{1,0}, \boldsymbol{p}_{2,0}, \ldots, \boldsymbol{p}_{2,q-2})$ WOM code, where $M_1 = 2^{R_1 n}$ and $M_2 = 2^{R_2 n}$ are determined as the maximal possible values based on ϵ, which tends to zero, and the probability vectors $\boldsymbol{p}_{j,i}$.

For the case $q=2$, for shorthand, given $p_{1,0 \to 1} = p$ we denote these codes by $[n, 2; M_1, M_2]^{EI:DU,z}(\epsilon, p)$ (where $p_{2,0 \to 1} = 0.5$). Furthermore, using cooling codes, the work in [1] provides the following family of binary WOM codes.

Lemma 8: For all $p \in [0, 0.5]$ and $\epsilon > 0$, there exists a two-write binary WOM code $[n, 2; M_1, M_2]^{EI:DU,z}(\epsilon, p)$ such that $M_1 = \sum_{i=0}^{\tau} \binom{n}{i}$ and $M_2 = 2^{n-\tau-1}$, where $\tau = pn$. Therefore, for any $\epsilon > 0$, there exists n such that $R_1 = \frac{\log M_1}{n} \geqslant h(p) - \epsilon$ and $R_2 = \frac{\log M_2}{n} \geqslant 1 - p - \epsilon$.

We are now ready to present a construction of two-change three-write EIA:DU-ELM codes which establishes the result in Theorem 5.

Construction 9: Given $p_{1,0}, p_{2,0}, p_{2,1} \in [0, 0.5]$ and $\epsilon > 0$, we construct an $[n, 3, 2; M_1, M_2, M_3]^{EIA:DU,z}$ ELM code where $M_j = 2^{n R_j}$ for $j = 1, 2, 3$ such that $R_1 \geqslant h(p_{1,0}) - \epsilon$, $R_2 \geqslant (1-p_{1,0}) h(p_{2,0}) + p_{1,0} h(p_{2,1}) - \epsilon$, and $R_3 \geqslant (1 - p_{1,0} p_{2,1}) - \epsilon$. We use the following two WOM codes.

1) Let $\boldsymbol{p}_{1,0} = (p_{1,0 \to 0}, p_{1,0 \to 1}, p_{1,0 \to 2}) = (1 - p_{1,0}, p_{1,0}, 0)$, $\boldsymbol{p}_{2,0} = (p_{2,0 \to 0}, p_{2,0 \to 1}, p_{2,0 \to 2}) = (0, p_{2,0}, 1 - p_{2,0})$, and $\boldsymbol{p}_{2,1} = (p_{2,1 \to 1}, p_{2,1 \to 2}) = (1 - p_{2,1}, p_{2,1})$. Let C_1 be an $[n, 2; M_1, M_2]_3^{EI:DU,z}(\epsilon, \boldsymbol{p}_{1,0}, \boldsymbol{p}_{2,0}, \boldsymbol{p}_{2,1})$ two-write

[1] We assume here that p_i is a rational number and n is large enough such that $p_i n$ is an integer for $i \in [q]$.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

ternary EI:DU WOM code from Lemma 7 with two pairs of encoder/decoder $(\mathcal{E}_{3,1}, \mathcal{D}_{3,1})$ and $(\mathcal{E}_{3,2}, \mathcal{D}_{3,2})$.

2) Let $\rho_1 = p_{1,0}p_{2,1}$, and C_2 be an $[n, 2; M_1', M_3]^{EI:DU,z}(\epsilon, \rho_1)$ two-write binary EI:DU WOM code from Lemma 8 with two pairs of encoder/decoder $(\mathcal{E}_{2,1}, \mathcal{D}_{2,1})$ and $(\mathcal{E}_{2,2}, \mathcal{D}_{2,2})$.

The three pairs of encoder/decoder mappings $(\mathcal{E}_j^{EIA:DU}, \mathcal{D}_j^{EIA:DU})$ for $j = 1, 2, 3$ are defined as follows.

First write: $\mathcal{E}_1^{EIA:DU}(m_1) = \mathcal{E}_{3,1}(m_1)$ for all $m_1 \in [M_1]$. Similarly, $\mathcal{D}_1^{EIA:DU}(\boldsymbol{c}_1) = \mathcal{D}_{3,1}(\boldsymbol{c}_1)$. Note that since we chose the probability to program level 2 in the first write of C_1 to be zero, the output of the encoder $\mathcal{E}_{3,1}$ is indeed a binary vector, so $\mathcal{E}_1^{EIA:DU}$ and $\mathcal{D}_1^{EIA:DU}$ are well defined.

Second write: The idea is to use the second write encoder $\mathcal{E}_{3,2}$ of C_1 with the probability vectors $\boldsymbol{p}_{2,0}$ and $\boldsymbol{p}_{2,1}$, and notice that here we write all cells to levels 1 or 2. Then, we can view this "ternary word" as a binary word. Let $\boldsymbol{c}_1 = (c_{1,1}, \ldots, c_{1,n}) \in Im(\mathcal{E}_1^{EIA:DU})$ be the cell-state vector after the first write, and note that this is a binary vector. The encoder/decoder $(\mathcal{E}_2^{EIA:DU}, \mathcal{D}_2^{EIA:DU})$ are defined formally as follows. For all $(m_2, \boldsymbol{c}_1) \in [M_2] \times Im(\mathcal{E}_1^{EIA:DU})$,

$$\boldsymbol{c}_2 = \mathcal{E}_2^{EIA:DU}(m_2, \boldsymbol{c}_1) = \boldsymbol{c}_2' (\mathrm{mod} 2),$$

where $\boldsymbol{c}_2' = \mathcal{E}_{3,2}(m_2, \boldsymbol{c}_1) \in [3]^n$. Furthermore, for all $\boldsymbol{c}_2 \in Im(\mathcal{E}_2^{EIA:DU})$,

$$\mathcal{D}_2^{EIA:DU}(\boldsymbol{c}_2) = \mathcal{D}_{3,2}(\boldsymbol{c}_2') = m_2,$$

where $\boldsymbol{c}_2' = 2 \cdot \mathbf{1} - \boldsymbol{c}_2$, that is, $c_{2,i}' = 1$ if $c_{2,i} = 1$ and $c_{2,i}' = 2$ if $c_{2,i} = 0$.

Third write: Let $\boldsymbol{c}_2$ be the cell-state vector after the second write. We note that the encoder on the third write knows the program-count vector $\boldsymbol{v}_2 \in [3]^n$, but the decoder does not have this information. Among the n cells, there are $\rho_1 n$ cells which have been programmed twice, where $\rho_1 = p_{1,0}p_{2,1}$, and therefore (only) these cells cannot be programmed on this write. Hence, the encoder can interpret the vector $\boldsymbol{v}_2$ as a length-n binary vector indicating for each cell whether it can be programmed on this write. We denote this vector by $\boldsymbol{c}_2''$, so $\boldsymbol{c}_{2,i}'' = 1$ if and only if $\boldsymbol{v}_{2,i} = 2$. We will use the code C_2 to encode and decode on this write, and we denote by $\bar{\boldsymbol{c}}$ the bitwise complement of a binary vector $\boldsymbol{c}$. Specifically, the encoder/decoder mappings are defined as follows. For all $m_3 \in [M_3]$ and $\boldsymbol{v}_2 \in N_2$,

$$\mathcal{E}_3^{EIA:DU}(m_3, \boldsymbol{v}_2) = \overline{\mathcal{E}_{2,2}(m_3, \boldsymbol{c}_2'')}.$$

Furthermore, for all $\boldsymbol{c}_3 \in Im(\mathcal{E}_3^{EIA:DU})$,

$$\mathcal{D}_3^{EIA:DU}(\boldsymbol{c}_3) = \mathcal{D}_{2,2}(\overline{\boldsymbol{c}_3}).$$

To illustrate Construction 9, we present the following example.

Example 1: Let $n = 7, p_{1,0} = 3/7, p_{2,0} = 1/2$, and $p_{2,1} = 1/3$, we construct a $[7, 3, 2; M_1, M_2, M_3]^{EIA:DU,z}$ three-change two-write ELM code as follows.

- In the first write, we encode a message m_1 to obtain a binary vector of length 7, e.g., $\boldsymbol{c}_1 = (1, 1, 1, 0, 0, 0, 0)$.
- In the second write, to encode a message m_2, in the first step, we use the second write encoder $\mathcal{E}_{3,2}$ of the ternary code C_1 in Lemma 7 with probability $\boldsymbol{p}_{2,0} = (0, 1/2, 1/2)$ and $\boldsymbol{p}_{2,1} = (2/3, 1/3)$ to obtain $\boldsymbol{c}_2' = \mathcal{E}_{3,2}(\boldsymbol{c}_1, m_2)$, e.g., $\boldsymbol{c}_2' = (2, 1, 1, 1, 1, 2, 2)$. In the second step, we can replace symbol 2 by symbol 0 in the vector $\boldsymbol{c}_2'$ to obtain the binary vector $\boldsymbol{c}_2 = (0, 1, 1, 1, 1, 0, 0)$. So, $\boldsymbol{c}_2$ is the output in the second write. We observe that it is not difficult to decode the vector $\boldsymbol{c}_2$ to obtain the message m_2.
- In the last write, the encoder has all information and know that the first cell is programmed twice and the program-count vector is $\boldsymbol{v}_2 = (2, 1, 1, 1, 1, 0, 0)$. The encoder also view the vector $\boldsymbol{v}_2$ as a binary vector $\boldsymbol{c}_2'' = (1, 0, 0, 0, 0, 0, 0)$ where the first cell (=1) is not programmable. Using the second-write encoder $\mathcal{E}_{2,2}$ of the EIU:DI WOM code C_2, we can encode the message m_3 to obtain $\boldsymbol{c}_3' = \mathcal{E}_{2,2}(m_3, \boldsymbol{c}_2'')$, e.g., $\boldsymbol{c}_3' = (1, 0, 0, 0, 1, 1, 1)$. We now take the bitwise complement of $\boldsymbol{c}_3'$ to obtain $\mathcal{E}_3^{EIA:DU}(m_3, \boldsymbol{v}_2) = \boldsymbol{c}_3 = (0, 1, 1, 1, 0, 0, 0)$. So, all the three messages are $\boldsymbol{c}_1 = (1, 1, 1, 0, 0, 0, 0), \boldsymbol{c}_2 = (0, 1, 1, 1, 1, 0, 0)$, and $\boldsymbol{c}_3 = (0, 1, 1, 1, 0, 0, 0)$.

□

We now present the proof of Theorem 5.

Proof of Theorem 5: Let $R_j(C_i)$ be the rate of the WOM code C_i on the j-th write. For any $\epsilon > 0$ and $p_{1,0}, p_{2,0}, p_{2,1} \in [0, 0.5]$, we choose the codes C_1 and C_2 in Construction 9 to satisfy

$$R_1(C_1) \geqslant h(\boldsymbol{p}_{1,0}) - \epsilon = h(p_{1,0}) - \epsilon,$$

where $\boldsymbol{p}_{1,0} = (1 - p_{1,0}, p_{1,0}, 0)$.

$$\begin{aligned} R_2(C_1) &\geqslant p_{1,0\to 0} h(\boldsymbol{p}_{2,0}) + p_{1,0\to 1} h(\boldsymbol{p}_{2,1}) - \epsilon \\ &\geqslant (1 - p_{1,0}) h(p_{2,0}) + p_{1,0} h(p_{2,1}) - \epsilon, \end{aligned}$$

and

$$R_2(C_2) \geqslant 1 - \rho_1 - \epsilon = 1 - p_{1,0}p_{2,1} - \epsilon.$$

The result follows from the fact that the rate tuple of the two-change three-write ELM code is $(R_1(C_1), R_2(C_1), R_2(C_2))$. ■

The solution for the case $t = 3, \ell = 2$ is generalized for any t and ℓ in the following theorem.

Theorem 10: For all t and ℓ, $\mathcal{C}_{t,\ell} \subseteq \mathcal{C}_{t,\ell}^{EIA:DU,z}$, that is, for any $\epsilon > 0$ and a rate t-tuple $(R_1, \ldots, R_t) \in \mathcal{C}_{t,\ell}$, there exists a zero-error ℓ-change t-write EIA:DU ELM code C such that its rate on the j-th write is at least $R_j - \epsilon$ for all $1 \leqslant j \leqslant t$, that is, $R_j(C) \geqslant R_j - \epsilon$.

To prove Theorem 10, we construct a zero-error ℓ-change t-write ELM code. The idea is to generalize Construction 9. Hence, for any given j, we use q-ary EI:DU WOM code from Lemma 7 to program all cells up to the two highest levels $q - 1$ and $q - 2$. So, the decoder can look at $q - 1$ as 0 and $q - 2$ as 1 to decode the original message. We now present the construction formally as follows.

Construction 11: Given $p_{j,i} \in [0, 0.5]$, for all $i \in [\ell + 1]$, $1 \leqslant j \leqslant t$, and $\epsilon > 0$, we construct an $[n, t, \ell; M_1, \ldots, M_t]^{EIA:DU,z}$ ELM code where $M_1 = \binom{n}{p_{1,0}n}$ and $M_j = 2^{nR_j}$ for $2 \leqslant j \leqslant t$ such that $R_j \geqslant \sum_{i=0}^{m-1} Q_{j-1,i} h(p_{j,i}) - \epsilon$, where $Q_{j-1,i}$ is defined

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

in Equation (1). The t pairs of encoder/decoder mappings $(\mathcal{E}_j^{EIA:DU}, \mathcal{D}_j^{EIA:DU})$ are defined as follows.

First write: Given $p_{1,0}$, we program all words of length n, weight $p_{1,0}n$ as on the first write of Construction 9. Hence, $M_1 = \binom{n}{p_{1,0}n}$ and the rate on the first write satisfies $R_1 \geqslant h(p_{1,0}) - \epsilon$.

j-th write, $2 \leqslant j \leqslant t$**:** Let $m = \min\{j, \ell\}$. We denote the cell-state vector and the cell-program-count vector after the $j-1$ writes by $\boldsymbol{c}_{j-1} = (c_{j-1,1}, \ldots, c_{j-1,n}) \in Im(\mathcal{E}_{j-1}^{EIA:DU})$ and $\boldsymbol{v}_{j-1} = (v_{j-1,1}, \ldots, v_{j-1,n}) \in \mathcal{B}(n, \boldsymbol{q}_{j-1}) \subset [m]^n$, respectively, where $\boldsymbol{q}_{j-1} = (Q_{j-1,0}, Q_{j-1,1}, \ldots, Q_{j-1,m-1})$ and $Q_{j,i}$ are defined in Equation (1). To program on the j-th write, we use the two-write $(2m+1)$-ary WOM code from Lemma 7, $[n, 2; M_{1,j}, M_{2,j}]_{2m+1}^{EI:DU,z}(\epsilon, \boldsymbol{p}_{1,0}, \boldsymbol{p}_{2,0}, \boldsymbol{p}_{2,1}, \ldots, \boldsymbol{p}_{2,2m-1})$ where $\boldsymbol{p}_{1,0} = (\boldsymbol{q}_{j-1}, 0, 0, \ldots, 0)$ and for all $i \in [m-1]$, $\boldsymbol{p}_{2,i} = (p_{2,i\to i}, \ldots, p_{2,i\to 2m}) = (0, \ldots, 0, p_{j,i}, 1-p_{j,i})$ if i is even and $\boldsymbol{p}_{2,i} = (p_{2,i\to i}, \ldots, p_{2,i\to 2m}) = (0, \ldots, 0, 1-p_{j,i}, p_{j,i})$ if i is odd. As in Lemma 7, $M_{2,j} = 2^{R_j n}$ where $R_j \geqslant \sum_{i=0}^{m-1} Q_{j-1,i} h(\boldsymbol{p}_{2,i}) - \epsilon = \sum_{i=0}^{m-1} Q_{j-1,i} h(p_{j,i}) - \epsilon$ since $\boldsymbol{p}_{2,i} = (p_{2,i\to i}, \ldots, p_{2,i\to 2m}) = (0, \ldots, 0, 1-p_{j,i}, p_{j,i})$ or $\boldsymbol{p}_{2,i} = (p_{2,i\to i}, \ldots, p_{2,i\to 2m}) = (0, \ldots, 0, p_{j,i}, 1-p_{j,i})$. Hence, on the j-th write, we choose $M_j = M_{2,j} = 2^{R_j n}$. We denote the two pairs of the encoder/decoder of the used WOM code by $(\mathcal{E}_{m,1}, \mathcal{D}_{m,1})$ and $(\mathcal{E}_{m,2}, \mathcal{D}_{m,2})$. The idea is to push all cells to the two highest levels and view the obtained word as a binary word. Hence, to decode correctly, the decoder only needs to know the cell-state vector after the j-th write which is a binary word. We now define the encoder/decoder $(\mathcal{E}_j^{EIA:DU}, \mathcal{D}_j^{EIA:DU})$ formally as follows. For all each $m_j \in [M_j]$ and $\boldsymbol{v}_{j-1} \in Im(\mathcal{E}_{m,1})$

$$\boldsymbol{c}_j = \mathcal{E}_j^{EIA:DU}(m_j, \boldsymbol{v}_{j-1}) = \boldsymbol{c}'_j (\mathrm{mod} 2),$$

where $\boldsymbol{c}'_j = \mathcal{E}_{m,2}(m_j, \boldsymbol{v}_{j-1}) \in [2m+1]^n$. Furthermore, for all $\boldsymbol{c}_j \in Im(\mathcal{E}_j^{EIA:DU})$,

$$\mathcal{D}_j^{EIA:DU}(\boldsymbol{c}_j) = \mathcal{D}_{m,2}(\boldsymbol{c}'_j) = m_j,$$

where $c'_{j,i} = 2m-1$ if $c_{j,i} = 1$ and $c'_{j,i} = 2m$ if $c_{j,i} = 0$.

Proof of Theorem 10: Given all parameters as in Construction 11, the rate of this ELM code on the first write is $R_1 \geqslant h(p_{1,0}) - \epsilon$. Now, we consider the j-th write. Since we used the WOM code in Lemma 7 to program the j-th write of the ELM code, the rate on this write is exactly the rate on the second write of the used WOM code. Hence, the rate in the j-th write of the ELM code is $R_j \geqslant \sum_{i=0}^{m-1} Q_{j-1,i}\, h(p_{j,i}) - \epsilon$. ∎

Remark 1: In this section, we provide an explicit construction of zero-error two-change three-write EIA:DU ELM code and generalize the result to construct a zero-error ℓ-change t-write EIA:DU ELM code. Since Shpilka [21] provided a pair of polynomial time encoding/decoding algorithms of a family of two-write WOM codes, the encoder and decoder in Theorem 10 also run in polynomial time. As shown in Theorem 10, using these constructions, we can achieve any rate in the capacity region and thus achieve the maximum sum-rate when the length n tends to infinity. However, for a fixed value of n, we can only achieve a high sum-rate but can not achieve the maximum sum-rate. Furthermore, Shpilka's technique only works for large block length [21]. Hence, for small value of block length n, we need other constructions to obtain a high sum-rate, for example, Construction 19 that will be presented later.

B. The EIA:DIA Model - Converse Part of Theorem 2

In this section, we prove the converse part of Theorem 2 for the EIA:DIA model ϵ-error case. That is, we prove that $\mathcal{C}_{t,\ell}^{EIA:DIA,\epsilon} \subseteq \mathcal{C}_{t,\ell}$.

For this direction we need to prove that if there exists an $[n, t, \ell; M_1, \ldots, M_t]^{EIA:DIA, \boldsymbol{p}_e}$ ELM code where $\boldsymbol{p}_e = (p_{e_1}, \ldots, p_{e_t})$, then

$$\left(\frac{\log M_1}{n} - \epsilon_1, \frac{\log M_2}{n} - \epsilon_2, \ldots, \frac{\log M_t}{n} - \epsilon_t\right) \in \mathcal{C}_{t,\ell},$$

where $(\epsilon_1, \epsilon_2, \ldots, \epsilon_t)$ tends to $\mathbf{0}$ if $\boldsymbol{p}_e$ tends to $\mathbf{0}$ and n tends to infinity. In our proof $\epsilon_j = \frac{H(p_{e_j}) + p_{e_j} \log(M_j)}{n}$, and therefore $\epsilon_j \to 0$ when $p_{e_j} \to 0$ and $n \to \infty$.

Let X_j be a length-n binary vector where $X_{j,k} = 1$ if and only if the k-th cell is intended to be programmed on the j-th write. Similarly, Y_j, is a length-n binary vector, where $Y_{j,k} = 1$ if and only if the value of the k-th cell was successfully changed on the j-th write, that is, $Y_j = \boldsymbol{c}_j \oplus \boldsymbol{c}_{j-1}$. Note that the encoder knows the number of times each cell was programmed. Therefore, we can assume that a cell is not intended to be programmed more than ℓ times. Furthermore, the decoder also knows the number of times each cell was programmed. Thus we assume that $X_j = Y_j$ where X_j is the encoded word and Y_j is the input of the decoder.

Let $S_1, \ldots, S_t$ be independent random variables, where S_j is uniformly distributed over the messages set $[M_j]$, and $\hat{S}_j$ is the decoding result on the j-th write. Let V_j be an independent random variable on N_j, the set of all cell-programs-count vectors after the first j writes. The data processing yields the following Markov chain:

$$S_j | V_{j-1} \text{ — } X_j | V_{j-1} \text{ — } Y_j | V_{j-1} \text{ — } \hat{S}_j | V_{j-1}$$

and therefore, $I(X_j; Y_j | V_{j-1}) \geqslant I(S_j; \hat{S}_j | V_{j-1})$.

Additionally,

$$\begin{aligned} I(S_j; \hat{S}_j | V_{j-1}) &= H(S_j | V_{j-1}) - H(S_i | \hat{S}_j, V_{j-1}) \\ &\geqslant H(S_j) - H(S_j | \hat{S}_j) \\ &\geqslant \log(M_j) - H(p_{e_j}) - p_{e_j} \log(M_j). \end{aligned}$$

The first inequality follows from the independence of V_{j-1} and S_j which implies that $H(S_j | V_{j-1}) = H(S_j)$, and from the fact that conditioning does not increase the entropy. The second inequality follows from Fano's inequality [7, p. 38] $H(S_j | \hat{S}_j) \leqslant H(p_{e_j}) + p_{e_j} \log(M_j)$.

Let L be an index random variable, which is uniformly distributed over the index set $[n]$. Since L is independent of

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

all other random variables we get

$$\begin{aligned}\frac{1}{n}I(X_j;Y_j|V_{j-1}) &\leqslant \frac{1}{n}H(Y_j|V_{j-1})\\ &\overset{(a)}{\leqslant} \frac{1}{n}\sum\nolimits_{k=0}^{n-1} H(Y_{j,k}|V_{j-1,k})\\ &\overset{(b)}{=} H(Y_{j,L}|V_{j-1,L},L)\\ &\overset{(c)}{\leqslant} H(Y_{j,L}|V_{j-1,L})\\ &= \sum\nolimits_{i=0}^{\ell} Pr(V_{j-1,L}=i)H(Y_{j,L}|V_{j-1,L}=i)\\ &\overset{(d)}{=} \sum\nolimits_{i=0}^{\ell-1} Pr(V_{j-1,L}=i)H(Y_{j,L}|V_{j-1,L}=i),\end{aligned}$$

where steps (a) and (c) follow from the fact that entropy of a vector is not greater than the sum of the entropies of its components, and conditioning does not increase the entropy. Step (b) follows from the fact that

$$\begin{aligned}H(Y_{j,L}|V_{j-1,L},L) &= \sum\nolimits_{k=0}^{n-1} Pr(L=k)H(Y_{j,k}|V_{j-1,L},L=k)\\ &= \frac{1}{n}\sum\nolimits_{k=0}^{n-1} H(Y_{j,k}|V_{j-1,k}),\end{aligned}$$

and step (d) follows from $H(Y_{j,L}|V_{j-1,L}=\ell)=0$.

Now, we set $p_{j,i} = Pr(X_{j,L}=1|N_{j-1,L}=i) = Pr(Y_{j,L}=1|N_{j-1,L}=i)$, and thus we can conclude that $Q_{j,i}=Pr(N_{j,L}=i)$ where $Q_{j,i}$ is calculated in Equation (1), and then

$$\begin{aligned}\frac{\log(M_j)}{n}-\epsilon_j &\leqslant \frac{1}{n}I(X_j;Y_j|N_{j-1})\\ &\leqslant \sum\nolimits_{i=0}^{\ell-1} Pr(N_{j-1,L}=i)H(Y_{j,L}|N_{j-1,L}=i)\\ &= \sum\nolimits_{i=0}^{\ell-1} Q_{j-1,i}h\left(p_{j,i}\right),\end{aligned}$$

where $\epsilon_j = \frac{H(p_{e_j})+p_{e_j}\log(M_j)}{n}$, and the converse part is implied.

By Theorem 10 in Subsection III-A and by the proof of the converse part in Subsection III-B we completed the proof of Theorem 2. Furthermore by Theorem 4 we conclude the following corollary.

Corollary 12: For all t and ℓ, $\mathcal{C}_{t,\ell}=\widehat{\mathcal{C}}_{t,\ell}$ is the capacity region for all the EIA models for both the zero-error and the ϵ-error cases and is denoted by $\mathcal{C}_{t,\ell}^{EIA}$. The maximum sum-rate of all the EIA models is $\mathcal{R}_{t,\ell}^{EIA}=\log\sum_{i=0}^{\ell}\binom{t}{i}$.

IV. The Capacity of the EIP:DIA Model

In this section we discuss the capacity region and the maximum sum-rate of the EIP:DIA model. Recall that if $\ell=1$ then by definition, EIP is equivalent to EIA and this model is equivalent to the known WOM model. Thus, in this section we assume that $\ell>1$. We focus on the ϵ-error case and present the capacity region of this model. The zero-error case is harder to solve, and is left for future research. However the ϵ-error case provides an upper bound for the zero-error case. Note that the EU:DI WOM model is simpler than the EIP:DIA ELM model, and even though its exact capacity for the zero-error case is still not known for general t. As done in the EIA models, let us denote by $\boldsymbol{c}_j$, $j\in[t+1]$, the length-n binary vector which represents the memory state after the j-th write, where $\boldsymbol{c}_0=\mathbf{0}$.

For $1\leqslant j\leqslant t$ and $i\in[\ell+1]$, we define the probabilities $p_{j,0}$, $p_{j,1}$, and $Q_{j,i}$ as follows. $p_{j,k}$ is the probability of programming a cell on the j-th write given that the value of this cell was k, $k\in\{0,1\}$, and $Q_{j,i}$ is the probability of a cell to be programmed exactly i times after the first j writes. Additionally, let $Q_{j,e},Q_{j,o}$ be the probability of a cell to be programmed an even, odd number of times after the first j writes, respectively. Formally, $Q_{j,i}$ is defined recursively by using the probabilities $p_{j',0}$ and $p_{j',1}$ for $j'\leqslant j$. We now assume that ℓ is even. The case of an odd ℓ is defined similarly. We define $Q_{j,i}$ for $j>0$ as follows. For even $i\geqslant 0$,

$$Q_{j,i}=\begin{cases}Q_{j-1,i-1}p_{j,1}+Q_{j-1,i}(1-p_{j,0}), & \text{if } 0<i<\ell,\\ Q_{j-1,i-1}p_{j,1}+Q_{j-1,i}, & \text{if } i=\ell,\\ Q_{j-1,i}(1-p_{j,0}), & \text{if } i=0,\end{cases} \tag{4}$$

and for odd $i>0$, $Q_{j,i}=Q_{j-1,i-1}p_{j,0}+Q_{j-1,i}(1-p_{j,1})$. The base $j=0$, is $Q_{0,0}=1$ and $Q_{0,i}=0$ for $i>0$. Furthermore, let $Q_{j,e}=\sum_{i=0}^{\ell/2}Q_{j,2i}$ and $Q_{j,o}=\sum_{i=1}^{\ell/2}Q_{j,2i-1}$.

Next, we define the rates region $\widetilde{\mathcal{C}}_{t,\ell}$ which will be proved to be the capacity region of the EIP:DIA model for the ϵ-error case. We present here the definition for even ℓ, while the odd case is defined similarly.

$$\begin{aligned}\widetilde{\mathcal{C}}_{t,\ell}=\Big\{&(R_1,R_2,\ldots,R_t)|\forall 1\leqslant j\leqslant t:\\ &R_j\leqslant Q_{j-1,o}h(p_{j,1})+(Q_{j-1,e}-Q_{j-1,\ell})h(p_{j,0}),\\ &p_{j,0},p_{j,1}\in[0,0.5] \text{ and } Q_{j,e},Q_{j,o},Q_{j,\ell} \text{ are defined above}\Big\}.\end{aligned} \tag{5}$$

For example, for $t=3,\ell=2$, we have that

$$\begin{aligned}\widetilde{\mathcal{C}}_{3,2}=\mathcal{C}_{3,2}=\Big\{&(R_1,R_2,R_3)|R_1\leqslant h(p_{1,0}),\\ &R_2\leqslant 1-p_{1,0}+p_{1,0}h(p_{2,1}),\\ &R_3\leqslant 1-p_{1,0}p_{2,1}, \text{and } p_{1,0},p_{2,1}\in[0,0.5]\Big\},\end{aligned}$$

which is achieved by substituting $p_{3,0}=p_{3,1}=p_{2,0}=0.5$ in Equations (2) and (5). Using the region $\widetilde{\mathcal{C}}_{t,\ell}$, the next theorem characterizes the capacity region of the EIP models for the ϵ-error case.

Theorem 13: The rates region $\widetilde{\mathcal{C}}_{t,\ell}$ is the capacity region of t-write ℓ-change ELM EIP:DIA model for the ϵ-error case. That is, $\widetilde{\mathcal{C}}_{t,\ell}=\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$.

Proof: To show the achievable region, we should prove that for each $\epsilon>0$ and $(R_1,R_2,\ldots,R_t)\in\widetilde{\mathcal{C}}_{t,\ell}$, there exists an $[n,t;M_1,\ldots,M_t]_{t,\ell}^{EIP:DIA,\boldsymbol{p}_e}$ ELM code, where for all $1\leqslant j\leqslant t$, $\frac{\log M_j}{n}\geqslant R_j-\epsilon$ and $\boldsymbol{p}_e=(p_{e_1},\ldots,p_{e_t})\leqslant(\epsilon,\ldots,\epsilon)$. We use the well-known random channel-coding theorem [7, p. 200] on each write. We describe the encoding and decoding on each write.

The j-th write presents a DMC with the input length-n binary vector X_j and the output is (Z_{j-1},Y_j), where

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

$Z_{j-1} \in [\ell+1]^n$ represents the times each cell was programmed before the j-th write, and $Y_j \in [2]^n$ represent the state of the memory after the j-th write. Let $x_j = X_{j,k}$, $z_j = Z_{j-1,k}$, and $y_j = Y_{j,k}$ for some index k. By the random coding theorem, for n large enough, the following region is achievable

$$\Big\{(R_1,\ldots,R_t)| \quad \forall 1 \leqslant j \leqslant t, R_j \leqslant I(x_j;y_j)\Big\}.$$

By the definitions and notations of the probabilities $p_{j',i'}$ and $Q_{j',i'}$,

$$\begin{aligned}
I(x_j;(z_{j-1},y_j)) &= H(z_{j-1},y_j) - H(z_{j-1},y_j|x_j) \\
&= H(z_{j-1}) + H(y_j|z_{j-1}) - H(z_{j-1},y_j|x_j) \\
&\overset{(a)}{=} H(z_{j-1}) + H(y_j|z_{j-1}) - H(z_{j-1}) \\
&= H(y_j|z_{j-1}) \\
&= \sum_{i=0}^{\ell} Pr(z_{j-1}=i) H(y_j|z_{j-1}=i) \\
&\overset{(b)}{=} \sum_{i=0}^{\ell-1} Pr(z_{j-1}=i) H(y_j|z_{j-1}=i) \\
&= \sum_{i=1}^{\ell/2} (Q_{j-1,2i-1} h(p_{j,1}) + Q_{j-1,2i-2} h(p_{j,0})) \\
&= Q_{j-1,o} h(p_{j,1}) + (Q_{j-1,e} - Q_{j-1,\ell}) h(p_{j,0}).
\end{aligned}$$

Step (a) follows from $H((z_{j-1},y_j)|x_i) = H(z_{j-1}|x_j)$ since y_j is a function of x_j, z_{j-1}, and $H(z_{j-1}|x_j) = H(z_{j-1})$ because z_{j-1} is independent on x_j. Step (b) is implied by $H(y_j|z_{j-1}=\ell) = 0$.

Hence, we can achieve the region $\widetilde{\mathcal{C}}_{t,\ell}$ for the ℓ-change t-write WℓM EIP:DIA model for the ϵ-error case.

The proof of the converse part, $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \subseteq \widetilde{\mathcal{C}}_{t,\ell}$, is similar to the proof of this part in Theorem 2, and hence is deferred to Appendix. ∎

We could also present a family of capacity achieving codes using the binary erasure channel (BEC). Note that on the j-th write, both the encoder and the decoder know $\boldsymbol{c}_{j-1}$, the state of the memory before writing the new data, while the decoder also knows $\boldsymbol{v}_{j-1}$, the number of times each cell was programmed before the j-th write. Therefore, the encoder on the j-th write treats the one and the zero cells separately. On the cells with value one, the encoder writes zero with probability $p_{j,1}$ (for example by using a constant weight code), while for the zero cells, the decoder knows which cells have been already programmed ℓ times before the j write. Thus, the encoding on the zero cells can be represented as encoding over BEC with erasure probability Q_ℓ/Q_e. The capacity of the BEC with erasure probability π and probability α for occurrence one in the encoded vector is $(1-\pi)h(\alpha)$ [7, p. 188]. By substituting $p_{j,0} = \alpha$ and $\pi = Q_\ell/Q_e$, we get the rate on the j-th write $Q_{j-1,o}h(p_{j,1}) + (Q_{j-1,e} - Q_{j-1,\ell})h(p_{j,0})$.

The following theorem is an immediate result deduced by the definitions of $\widetilde{\mathcal{C}}_{t,\ell}$ and $\mathcal{C}_{t,\ell}$ and Theorems 2 and 13.

Theorem 14: For $\ell = 2$ the capacity region of the EIP:DIA model for the epsilon error case is equal to the capacity region for the EIA models, i.e., $\mathcal{C}_{t,2}^{EIP:DIA,\epsilon} = \mathcal{C}_{t,2}^{EIA}$.

In Section V, we compare between the EIP:DIA model which was discussed in this section, and the EIA models, which were presented in Section III.

V. A Comparison Between the EIA Models and the EIP:DIA Model

In this section we compare between the EIA models and the EIP:DIA model. The capacity of the EIA models, $\mathcal{C}_{t,\ell}^{EIA:DY,g}$ for $g \in \{z,\epsilon\}$ and $Y \in \{IA, IP, U\}$, was stated in Section III to be equal to $\mathcal{C}_{t,\ell}$, while in Section IV we presented the capacity region of the EIP:DIA model for the ϵ-error case, $\widetilde{\mathcal{C}}_{t,\ell} = \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$.

The next theorem proves that for $t > \ell \geqslant 3$ the maximum sum-rate of the EIP:DIA model for the epsilon-error case is smaller than the maximum sum-rate of the EIA models. Hence, the capacity region $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$ is a proper subset of the capacity region $\mathcal{C}_{t,\ell}^{EIA}$ for these parameters. Recall that for $\ell = 2$ these regions were shown to be the same in Theorem 14, and therefore the maximum sum-rates of these models for $\ell = 2$ are the same too.

Theorem 15: For $t > \ell \geqslant 3$, $\mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIA}$, and hence $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIA}$.

Proof: Let $\widetilde{\mathbf{R}} = (\widetilde{R}_1, \widetilde{R}_2, \ldots, \widetilde{R}_t)$ be a rate tuple which achieves the maximum sum-rate $\mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon}$, and we denote by $\widetilde{p}_{j,0}$, $\widetilde{p}_{j,1}$, and $\widetilde{Q}_{j,i}$, $1 \leqslant j \leqslant t$ and $i \in [\ell+1]$, the probabilities which attain $\widetilde{\mathbf{R}}$ in $\widetilde{\mathcal{C}}_{t,\ell}$.

Now we present a rate tuple $\mathbf{R} = (R_1, R_2, \ldots, R_t) \in \mathcal{C}_{t,\ell} > \widetilde{\mathbf{R}}$. Then, we conclude that $\mathbf{R} \in \mathcal{C}_{t,\ell}^{EIA} \setminus \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$, which implies that $\mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIA}$ and $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIA}$.

We assume now that ℓ is even, while the proof for the odd case is similar. Since $\widetilde{\mathbf{R}}$ is maximal rate tuple we have $\widetilde{p}_{t-1,0} = \widetilde{p}_{t,0} = \widetilde{p}_{t,1} = 0.5$. For all j and i, $1 \leqslant j \leqslant t-2$ and $i \in [\ell]$, we define $p_{j,i} = \widetilde{p}_{j,i'}$ where $i' = i \mod 2$. In addition, for all $i \in [\ell-1]$, $p_{t-1,i} = 0.5$, $p_{t-1,\ell-1} = \widetilde{p}_{t-1,1}$, and for all i, $p_{t,i} = 0.5$.

Thus, for all j and i, $1 \leqslant j \leqslant t-2$ and $i \in [\ell]$, $R_j = \widetilde{R}_j$ and $Q_{j,i} = \widetilde{Q}_{j,i}$. For the $(t-1)$-th write we have, $R_{t-1} = 1 - \widetilde{Q}_{t-2,\ell-1} - \widetilde{Q}_{t-2,\ell} + \widetilde{Q}_{t-2,\ell-1} h(\widetilde{p}_{t-1,1})$ while $\widetilde{R}_{t-1} = \widetilde{Q}_{t-2,o} h(\widetilde{p}_{t-1,1}) + (\widetilde{Q}_{t-2,e} - \widetilde{Q}_{t-2,\ell})$, and for the last write $R_t = \widetilde{R}_t = 1 - \widetilde{Q}_{t-1,\ell}$.

Now we prove that $\widetilde{p}_{t-1,1} < 0.5$ which immediately implies that $R_{t-1} > \widetilde{R}_{t-1}$ and thus completes the proof. Recall that $\widetilde{R}_t = 1 - \widetilde{Q}_{t-1,\ell} = 1 - \widetilde{Q}_{t-2,\ell} - \widetilde{Q}_{t-2,\ell-1}\widetilde{p}_{t-1,1}$. Thus, given the probabilities for the first $t-2$ writes, in order to achieve the maximal rate tuple $\widetilde{\mathbf{R}}$, we have to maximize $\widetilde{R}_{t-1} + \widetilde{R}_t$. That is, we choose $\widetilde{p}_{t-1,1}$ which maximizes $\widetilde{Q}_{t-2,o} h(\widetilde{p}_{t-1,1}) - \widetilde{Q}_{t-2,\ell-1}\widetilde{p}_{t-1,1}$. The derivative is $\widetilde{Q}_{t-2,0} \log(\frac{1-\widetilde{p}_{t-1,1}}{\widetilde{p}_{t-1,1}}) - \widetilde{Q}_{t-2,\ell-1}$, and the maximum is obtained for $\widetilde{p}_{t-1,1} = 1/(1 + 2^{\widetilde{Q}_{t-2,\ell-1}/\widetilde{Q}_{t-2,o}})$. Since $\widetilde{\mathbf{R}}$ is maximal and $t > \ell \geqslant 3$, we have $\widetilde{Q}_{t-2,\ell-1} > 0$, and therefore $\widetilde{p}_{t-1,1} \neq 0.5$. ∎

We can summarize the results regarding the capacity region of the EIP:DIA model in the following corollary.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

Corollary 16: For all $t > \ell$ the following holds

$$\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} = \widetilde{\mathcal{C}}_{t,\ell} \subseteq \mathcal{C}_{t,\ell} = \mathcal{C}_{t,\ell}^{EIA}.$$

Furthermore,

- For $t > \ell = 2$ all these regions are equal, in particular, $\mathcal{C}_{t,2}^{EIP:DIA,\epsilon} = \mathcal{C}_{t,2}^{EIA}$.
- For $t > \ell \geqslant 3$, $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIA}$ and $\mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIA}$.

VI. The Capacity of EU:DIA Model

In this section we study the EU:DIA model for the ϵ-error case, and provide the capacity region of this model. As in the EIP:DIA model, the capacity region for the zero-error case and the exact maximum sum-rate are left for future research.

For $1 \leqslant j \leqslant t$ and $i \in [\ell+1]$, let p_j be the probability of programming a cell on the j-th write, and $Q_{j,i}$ denotes the probability of a cell to be programmed exactly i times on the first j writes. Additionally, let $Q_{j,e}, Q_{j,o}$ be the probabilities of a cell to be programmed even, odd number of times on the first j writes, respectively. Formally, $Q_{j,i}$ is defined recursively by using $p_{j'}$ probabilities for $j' \leqslant j$. For $j \geqslant 1$,

$$Q_{j,i} = \begin{cases} Q_{j-1,i-1}p_j + Q_{j-1,i}(1-p_j), & \text{if } 0 < i \leqslant \ell, \\ Q_{j-1,i-1}p_j + Q_{j-1,i}, & \text{if } i = \ell, \\ Q_{j-1,i}(1-p_j), & \text{if } i = 0, \end{cases} \tag{6}$$

where $Q_{0,0} = 1$ and $Q_{0,i} = 0$ for $i > 0$.

Then, we define the region $\overline{\mathcal{C}}_{t,\ell}$ which is proved later in this section to be the capacity region $\mathcal{C}_{t,\ell}^{EU:DIA,\epsilon}$.

$$\overline{\mathcal{C}}_{t,\ell} = \Big\{(R_1, R_2, \ldots, R_t) | \forall 1 \leqslant j \leqslant t : R_j \leqslant h(p_j) - Q_{j-1,\ell} h(p_j), \; p_j \in [0, 0.5], \; Q_{j,\ell} \text{ is defined above} \Big\}. \tag{7}$$

The next theorems establish the capacity region of the EU:DIA model for the ϵ-error case and compare between this model and the EIP:DIA model. The techniques applied for the EU:DIA model are very similar to the proofs in Section IV. The proofs of Theorems 17 and 18 are similar to the proofs of Theorems 13 and 15, respectively. Therefore, these proofs are moved to Appendix.

Theorem 17: The rates region $\overline{\mathcal{C}}_{t,\ell}$ is the capacity region of t-write ℓ-change ELM EU:DIA model for the ϵ-error case. That is, $\overline{\mathcal{C}}_{t,\ell} = \mathcal{C}_{t,\ell}^{EU:DIA,\epsilon}$.

Theorem 18: For $t > \ell \geqslant 2$, $\mathcal{R}_{t,\ell}^{EU:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon}$, and hence $\mathcal{C}_{t,\ell}^{EU:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$.

VII. The EIP:DU Model

In this section, we study the EIP:DU model and its sum-rate. First, we note that $\mathcal{C}_{t,\ell}^{EIP:DU,\epsilon} \subseteq \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$ for all t, ℓ, and thus,

$$\mathcal{R}_{t,\ell}^{EIP:DU,\epsilon} \leqslant \mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon} \leqslant \log \sum_{i=0}^{\ell} \binom{t}{i}. \tag{8}$$

That is, we obtain an upper bound of the maximum sum-rate $\mathcal{R}_{t,\ell}^{EIP:DU,\epsilon}$. Note that for $t > \ell \geqslant 3$ this upper bound is not tight (Theorem 15). We are now interested in some good lower bounds for the maximum sum-rate. Our goal is to provide several constructions with high sum-rate. We first present a general construction for the zero-error case and then show how to obtain higher sum-rate for the ϵ-error case with $t = 3, \ell = 2$.

The following construction provides a family of ℓ-change t-write EIP:DU ELM codes for the zero-error case.

Construction 19: Let $(k_1, \ldots, k_\ell)$ be such that $1 \leqslant k_i \leqslant t$ for $1 \leqslant i \leqslant \ell$ and $\sum_{i=1}^{\ell} k_i = t$. Let $[n, k_i; M_{j_i+1}, \ldots, M_{j_i+k_i}]^{EI:DU,z}$ be a binary k_i-write EI:DU WOM code for $1 \leqslant i \leqslant \ell$ with sum-rate R_i where $j_1 = 0$ and $j_i = \sum_{r=1}^{i-1} k_r$. Each of which consists of n bits and k_i pairs of encoding and decoding maps $(\mathcal{E}_{j_i+h}^{EI:DU}, \mathcal{D}_{j_i+h}^{EI:DU})$ for $1 \leqslant h \leqslant k_i$. We define an $[n, t, \ell; M_1, \ldots, M_t]^{EIP:DU,z}$ ℓ-change t-write ELM code consists of n bits and t pairs of encoders and decoders $(\mathcal{E}_j^{EIP:DU}, \mathcal{D}_j^{EIP:DU})$ where $\mathcal{E}_j^{EIP:DU} = \mathcal{E}_j^{EI:DU}$ and $\mathcal{D}_j^{EIP:DU} = \mathcal{D}_j^{EI:DU}$ for $1 \leqslant j \leqslant t$.

The maximum sum-rate of the ELM codes from Construction 19 is $R_{sum} \geqslant \sum_{i=1}^{\ell} \log(k_i + 1) - \epsilon$ since for $1 \leqslant i \leqslant \ell$, $R_i \geqslant \log(k_i + 1) - \epsilon/\ell$ and $R_{sum} = \sum_{i=1}^{\ell} R_i$. Hence, in order to maximize the sum-rate, our goal is to maximize the value of $\sum_{i=1}^{\ell} \log(k_i + 1)$ given that $\sum_{i=1}^{\ell} k_i = t$. Assume that $t = k\ell + r$, $r \in [\ell]$, then this maximum value will be achieved when choosing $k_1 = \cdots = k_r = k + 1$ and $k_{r+1} = \cdots = k_\ell = k$. The next corollary summarizes this result.

Corollary 20: For all t and ℓ, where $t = k\ell + r$, $r \in [\ell]$,

$$\mathcal{R}_{t,\ell}^{EIP:DU,z} \geqslant r \log(k+2) + (\ell - r)\log(k+1) = \ell \log\left(\left\lfloor \frac{t}{\ell} \right\rfloor + 1\right) + (t \bmod \ell) \log\left(1 + \frac{1}{\lfloor \frac{t}{\ell} \rfloor + 1}\right).$$

Proof: We choose $(k_1, \ldots, k_\ell)$ such that $k_1 = \cdots = k_r = k + 1$ and $k_{r+1} = \cdots = k_\ell = k$ and thus $\sum_{i=1}^{\ell} k_i = t$. We note that $k = \lfloor \frac{t}{\ell} \rfloor$ and $r = t \mod \ell$. Since we presented in Construction 19 an $[n, t, \ell; M_1, \ldots, M_t]^{EIP:DU,z}$ ℓ-change t-write ELM code with sum-rate $R_{sum} = \sum_{i=1}^{\ell} R_i \geqslant r \log(k+2) + (\ell - r)\log(k+1) - \epsilon$ for any $\epsilon > 0$, we obtain the result in Corollary 20. ∎

From the above corollary, we have a lower bound of the maximum sum-rate of the EIP:DU model. Recall that $\mathcal{R}_{t,\ell}^{EIP:DU,z} \leqslant \mathcal{R}_{t,\ell}^{EIA} = \log \sum_{i=0}^{\ell} \binom{t}{i}$, that is the exact maximum sum-rate of the EIA model is an upper bound of the maximum sum-rate of the EIP:DU model. Hence, we obtain a lower bound and an upper bound of the maximum sum-rate of the EIP:DU ELM model. We note that when $t \leqslant \ell$, we always achieve the full capacity, that is, the maximum sum-rate is t. When $t > \ell$, the maximum sum-rate is difficult to compute exactly and there is a gap between the above lower and upper bounds. We illustrate the results for $\ell = 2, t \in [3, 23]$ in the following figure.

The following result shows that for $\ell = 2$ the sum-rate of the ELM code from Construction 19 is already close to the upper bound when t is large and $n \to \infty$.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

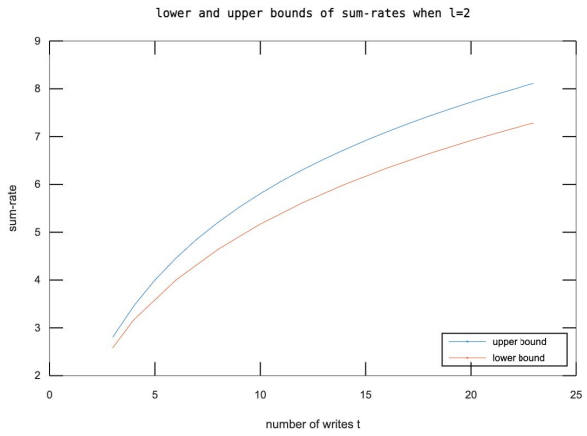

Fig. 1. The upper and lower bounds of the maximum sum-rates of the EIP:DU ELM codes when $\ell = 2, t \in [3, 23]$.

Proposition 21: For $\ell = 2$ and $t \geqslant 3$, $\mathcal{R}_{t,2}^{EIP:DU,z} \geqslant \mathcal{R}_{t,2}^{EIA} - 1$.

Proof: Recall that $\mathcal{R}_{t,2}^{EIA} = \log \sum_{i=0}^{2} \binom{t}{i} = \log \frac{t^2+t+2}{2}$. When t is even, there exists a positive integer t_1 such that $t = 2t_1$. In this case,

$$\mathcal{R}_{t,2}^{EIP:DU,z} \geqslant 2\log(t_1+1) = \log(t_1^2 + 2t_1 + 1)$$

and,

$$\mathcal{R}_{t,2}^{EIA} = \log \frac{4t_1^2 + 2t_1 + 1}{2}.$$

Hence,

$$\mathcal{R}_{t,2}^{EIA} - \mathcal{R}_{t,2}^{EIP:DU,z} \leqslant \log \frac{4t_1^2 + 2t_1 + 1}{2(t_1^2 + 2t_1 + 1)} \leqslant \log 2 = 1.$$

When t is odd, there exists a positive integer t_2 such that $t = 2t_2 + 1$. In this case,

$$\mathcal{R}_{t,2}^{EIP:DU,z} \geqslant \log(t_2+1) + \log(t_2+2) = \log(t_2^2 + 3t_2 + 2)$$

and,

$$\mathcal{R}_{t,2}^{EIA} = \log \frac{4t_2^2 + 6t_2 + 4}{2}.$$

Hence,

$$\mathcal{R}_{t,2}^{EIA} - \mathcal{R}_{t,2}^{EIP:DU,z} \leqslant \log \frac{4t_2^2 + 6t_2 + 4}{2(t_2^2 + 3t_2 + 2)} \leqslant \log 2 = 1.$$

In conclusion, the proposition is proven. ∎

We note that when $t = 3$ and $\ell = 2$, the maximum achievable sum-rate of the codes in Construction 19 is $\log 6 \approx 2.585$, while the upper bound is $\log 7 \approx 2.807$. Lastly, we show how to improve this result for the ϵ-error case.

The main ideas of the following construction are as follows. On the first two writes, we follow exactly the first two writes of Construction 9 which is a construction for a two-change three-write EIA:DU ELM code. After the second write, there are $\rho_1 n$ cells which were programmed twice, where $\rho_1 = p_{1,0}p_{2,1}$. However, while the encoder in the EIA:DU model knows these positions, the encoder in the third write in the EIP:DU model does not know these positions. In order to overcome this difficulty, we use the following family of binary EU:DU WOM codes.

Definition 22: An $[n, 2; M_1, M_2]_2^{EU:DU,(p_{e_1},p_{e_2})}(p_1, p_2)$ two-write binary EU:DU WOM code is a coding scheme comprising of n bits. It consists of two pairs of encoding and decoding maps $(\mathcal{E}_j^{EU:DU}, \mathcal{D}_j^{EU:DU})$ for $j = 1, 2$. For the map $\mathcal{E}_j^{EU:DU}$, $Im(\mathcal{E}_j^{EU:DU})$ is its image and $Im^*(\mathcal{E}_j^{EU:DU})$ is the set of all the cell-state vectors which can be obtained after the j-th write. We note that $Im(\mathcal{E}_0^{EU:DU}) = Im^*(\mathcal{E}_0^{EU:DU}) = \{(0, \ldots, 0)\}$ and $Im^*(\mathcal{E}_2^{EU:DU}) = \{\max\{\boldsymbol{c}_1, \boldsymbol{c}_2\}$ where $\boldsymbol{c}_i \in Im(\mathcal{E}_i^{EU:DU}) : i = 1, 2\}$. The encoding and decoding maps are defined as follows. For $j = 1, 2$,

$$\mathcal{E}_j^{EU:DU} : [M_j] \to \mathcal{B}(n, (1-p_j, p_j))$$

and

$$\mathcal{D}_j^{EU:DU} : Im^*(\mathcal{E}_j^{EU:DU}) \to [M_j]$$

such that for all $m \in [M_j]$,

$$\sum_{(m,\boldsymbol{c}) \in [M_j] \times Im^*(\mathcal{E}_{j-1}^{EU:DU})} Pr(m) Pr(\boldsymbol{c}) I_m\big(\mathcal{D}_j^{EU:DU}(\max\{\boldsymbol{c}, \mathcal{E}_j^{EU:DI}(m)\})\big) \leqslant p_{e_i}.$$

Two-write binary EU:DU WOM codes have been studied for a long time [23]. Recently, in [11] several constructions of EU:DU WOM codes were presented. Assume that there exists a capacity achieving code for the Z channel, then the following result for EU:DU WOM codes can be received based upon the constructions from [11].

Lemma 23: [11] For all $0 \leqslant p_1, p_2 \leqslant 0.5$ and $\epsilon > 0$ there exists an $[n, 2; M_1, M_2]^{EU:DU,(0,\epsilon)}$ two-write binary EU:DU WOM code satisfying:

- $\boldsymbol{c}_1 \in \mathcal{B}(n, (1-p_1, p_1))$, and $R_1 = \frac{\log M_1}{n} \geqslant h(p_1) - \epsilon$.
- $\boldsymbol{c}_2 \in \mathcal{B}(n, (1-p_2, p_2))$, and $R_2 = \frac{\log M_2}{n} \geqslant h(p_1 p_2) - p_2 h(p_1) - \epsilon$,

where $\boldsymbol{c}_i \in Im(\mathcal{E}_i^{EU:DU})$ for $i = 1, 2$.

We refer to the family of WOM codes from Lemma 23 as an $[n, 2; M_1, M_2]_q^{EU:DU,(0,\epsilon)}(\epsilon, p_1, p_2)$ WOM code, where $M_1 = 2^{R_1 n}$ and $M_2 = 2^{R_2 n}$ are determined as the maximal possible values based on ϵ, which tends to zero, and the probabilities p_1 and p_2.

We are now ready to present a construction of two-change three-write EIP:DU ELM code.

Construction 24: Given $p_{1,0}, p_{2,0}, p_{2,1}, p_3 \in [0, 0.5]$, we use the following two codes:

- An $[n, 3, 2; M_1, M_2, M_3']^{EIA:DU,z}$ code from Construction 9 with the first two pairs of encoder/decoder $(\mathcal{E}_i^{EIA:DU}, \mathcal{D}_i^{EIA:DU})$ for $i = 1, 2$.
- An $[n, 2; M_1', M_3]^{EU:DU,(0,\epsilon))}(\epsilon, \rho_1, p_3)$ two-write binary EU:DU WOM code from Lemma 23, $\rho_1 = p_{1,0}p_{2,1}$, with the pair of encoder/decoder in the second write $(\mathcal{E}_2^{EU:DU}, \mathcal{D}_2^{EU:DU})$.

We construct an $[n, 3, 2; M_1, M_2, M_3]^{EIP:DU,(0,0,\epsilon)}$ two-change three-write EIP:DU ELM code where its 3 pairs of encoding/decoding maps $(\mathcal{E}_j^{EIP:DU}, \mathcal{D}_j^{EIP:DU})$ for $j = 1, 2, 3$ are defined as follows.

(1) For $i = 1, 2$, $\mathcal{E}_i^{EIP:DU} = \mathcal{E}_i^{EIA:DU}$ and $\mathcal{D}_i^{EIP:DU} = \mathcal{D}_i^{EIA:DU}$. That is, the first two writes of this EIP:DU

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

ELM code are exactly the same as the first two writes of the EIA:DU ELM code from Construction 9.

(2) After the first two writes, we note that $\rho_1 n$ cells are already programmed twice, and thus can not be programmed this time. Hence, we use the pair of encoder/decoder $(\mathcal{E}_2^{EU:DU}, \mathcal{D}_2^{EU:DU})$ to encode/decode information. The pair of encoder/decoder in the third write is defined formally as follows:

$$\mathcal{E}_3^{EIP:DU} : [M_3] \times Im^*(\mathcal{E}_2^{EIP:DU}) \rightarrow [2]^n$$

such that for all $m_3 \in [M_3]$ and $\boldsymbol{c}_2 \in Im^*(\mathcal{E}_2^{EIP:DU})$, $\mathcal{E}_3^{EIP:DU}(m_3, \boldsymbol{c}_2) = \overline{\mathcal{E}_2^{EU:DU}(m_3)}$. Furthermore,

$$\mathcal{D}_3^{EIP:DU} : Im^*(\mathcal{E}_3^{EIP:DU}) \rightarrow [M_3]$$

such that for all $\boldsymbol{c}_3^* \in Im^*(\mathcal{E}_3^{EU:DU})$, $\mathcal{D}_3^{EIP:DU}(\boldsymbol{c}_3^*) = \mathcal{D}_2^{EU:DU}(\overline{\boldsymbol{c}_3^*}) = m_3$.

On the first two writes, it is clear that $R_1 \geqslant h(p_{1,0}) - \epsilon$ and $R_2 \geqslant (1-p_{1,0})h(p_{2,0}) + p_{1,0}h(p_{2,1}) - \epsilon$. In the third write, $R_3 \geqslant h(p_{1,0}p_{2,1}p_3) - p_3 h(p_{1,0}p_{2,1}) - \epsilon$.

In conclusion, we constructed a two-change three-write EIP:DU ELM code satisfying $R_1 \geqslant h(p_{1,0}) - \epsilon$, $R_2 \geqslant (1-p_{1,0})h(p_{2,0}) + p_{1,0}h(p_{2,1}) - \epsilon$ and $R_3 \geqslant h(p_{1,0}p_{2,1}p_3) - p_3 h(p_{1,0}p_{2,1}) - \epsilon$ for all $\epsilon > 0$.

Therefore, the following region is achievable for the ϵ-error case:

$$\begin{aligned} C_{3,2}^{EIP:DU} = \{(R_1, R_2, R_3) : {} & R_1 \leqslant h(p_{1,0}), \\ & R_2 \leqslant (1-p_{1,0})h(p_{2,0}) + p_{1,0}h(p_{2,1}), \\ & R_3 \leqslant h(p_{1,0}p_{2,1}p_3) - p_3 h(p_{1,0}p_{2,1}), \\ & p_{1,0}, p_{2,0}, p_{2,1}, p_3 \in [0,1]\}. \end{aligned}$$

The sum-rate of the above code is $R_{sum} = R_1 + R_2 + R_3 \geqslant h(p_{1,0}) + (1-p_{1,0})h(p_{2,0}) + p_{1,0}h(p_{2,1}) + h(p_{1,0}p_{2,1}p_3) - p_3 h(p_{1,0}p_{2,1}) - \epsilon$ for any $\epsilon > 0$. By choosing $p_{1,0} = 3/7, p_{2,0} = 1/2, p_{2,1} = 2/3$, and $p_3 = 1/2$, we obtain the sum-rate $R_{sum} = R_1 + R_2 + R_3 \approx 2.64$.

Remark 2: In this section, we construct a family of zero-error ℓ-change t-write EIP:DU ELM codes for any ℓ and t. Using some efficient encoding/decoding algorithms of the well-known binary t-write EI:DU WOM codes, we can encode/decode our EIP:DU ELM codes efficiently in polynomial time. When n tends to infinity, we can obtain some codes with high sum-rate and thus get a lower bound on the maximal sum-rate of the EIP:DU model. We note that the lower bound is not tight even though it is close to the upper bound. We actually improve the lower bound for the ϵ-error case when $\ell = 2$ and $t = 3$ in Construction 24. Using some known polynomial time encoding/decoding algorithms of a two-write EU:DU WOM code in Lemma 23 [11], the encoding and decoding algorithms in Construction 24 also run in polynomial time. Since the exact capacity region and the maximum sum-rate of the EIP:DU model are not known yet, we expect to have better constructions in near future.

VIII. Conclusion

In this paper, we have proposed and studied a new coding scheme, called ELM codes. This family of codes can be used to increase the endurance of resistive memories by rewriting codes. This new family of rewriting codes generalizes the well-known WOM codes. We investigated the coding schemes of nine different models which depend upon the knowledge of the encoder and the decoder. In all these models, we focused on the capacity region and the achievable maximum sum-rate. In several important models, we also presented constructions of ELM codes with high sum-rate and some constructions of capacity-achieving codes. For future work, we are interested in practical constructions of capacity-achieving codes with efficient encoding/decoding algorithms, especially in the EIP:DU model.

Appendix

Theorem 13 - The Converse Part: The rates region $\widetilde{\mathcal{C}}_{t,\ell}$ is a superset of the capacity region of t-write ℓ-change ELM EIP:DIA model for the ϵ-error case. That is, $\mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \subseteq \widetilde{\mathcal{C}}_{t,\ell}$.

Proof: Let S_j, $\hat{S}_j$, V_j, $1 \leqslant j \leqslant t$, and L be defined as in the proof of the converse part in Theorem 2. Thus, exactly as proved in Theorem 2, we have $I(X_j; Y_j | V_{j-1}) \geqslant I(S_j; \hat{S}_j | V_{j-1})$, $I(S_j; \hat{S}_j | V_{j-1}) \geqslant \log(M_j) - H(p_{e_j}) - p_{e_j}\log(M_j)$, and

$$\frac{1}{n} I(X_j; Y_j | V_{j-1}) \leqslant \sum_{i=0}^{\ell-1} Pr(V_{j-1,L} = i) H(Y_{j,L} | V_{j-1,L} = i).$$

Now, we set $p_{j,0} = Pr(X_{j,L} = 1 | V_{j-1,L} \mod 2 = 0)$ and similarly $p_{j,1} = Pr(X_{j,L} = 0 | V_{j-1,L} \mod 2 = 1)$. Thus, for even $i < \ell$ $H(Y_{j,L} | V_{j-1,L} = i) = H(p_{j,0})$, and for odd $i < \ell$ $H(Y_{j,L} | V_{j-1,L} = i) = H(p_{j,1})$. We also define for $i \in [\ell + 1]$ $Q_{j,i} = Pr(V_{j,L} = i)$, and we note that $Q_{j,i}$ can be calculated as in Equation (4), and we use the notations $Q_{j,o}$ and $Q_{j,e}$ as defined above. Then,

$$\begin{aligned} \frac{\log(M_j)}{n} - \epsilon_j \leqslant {} & \frac{1}{n} I(X_j; Y_j | V_{j-1}) \\ \leqslant {} & \sum_{i=0}^{\ell-1} Pr(V_{j-1,L} = i) H(Y_{j,L} | V_{j-1,L} = i) \\ = {} & \sum_{i=1}^{\ell/2} \left(Q_{j-1,2i-1} h\left(p_{j,1}\right) + Q_{j-1,2i-2} h\left(p_{j,0}\right) \right) \\ = {} & h(p_{j,1}) \sum_{i=1}^{\ell/2} Q_{j-1,2i-1} + h(p_{j,0}) \sum_{i=1}^{\ell/2} Q_{j-1,2i-2} \\ = {} & Q_{j-1,o} h(p_{j,1}) + (Q_{j-1,e} - Q_{j-1,\ell}) h\left(p_{j,0}\right), \end{aligned}$$

where $\epsilon_j = \frac{H(p_{e_j}) + p_{e_j}\log(M_j)}{n}$, and the claim is implied. ■

Theorem 17: The rates region $\overline{\mathcal{C}}_{t,\ell}$ is the capacity region of t-write ℓ-change ELM EU:DIA model for the ϵ-error case. That is, $\overline{\mathcal{C}}_{t,\ell} = \mathcal{C}_{t,\ell}^{EU:DIA,\epsilon}$.

Proof: To show the achievable region, we should prove that for each $\epsilon > 0$ and $(R_1, R_2, \ldots, R_t) \in \overline{\mathcal{C}}_{t,\ell}$, there exists an $[n, t; M_1, \ldots, M_t]_{t,\ell}^{EU:DIA,\boldsymbol{p}_e}$ ELM code, where for all $1 \leqslant j \leqslant t$, $\frac{\log M_j}{n} \geqslant R_j - \epsilon$ and $\boldsymbol{p}_e = (p_{e_1}, \ldots, p_{e_t}) \leqslant (\epsilon, \ldots, \epsilon)$. We use the well-known random channel-coding theorem [7, p. 200] on each write. We describe the encoding and decoding on each write.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

The j-th write presents a DMC with the input length-n binary vector X_j and the output is (Z_{j-1}, Y_j), where $Z_{j-1} \in [\ell+1]^n$ represents the times each cell was programmed before the j-th write, and $Y_j \in [2]^n$ represent the state of the memory after the j-th write. Let $x_j = X_{j,k}$, $z_j = Z_{j-1,k}$, and $y_j = Y_{j,k}$ for some index k. By the random coding theorem, for n large enough, the following region is achievable

$$\left\{(R_1, \ldots, R_t) | \quad \forall 1 \leqslant j \leqslant t, R_j \leqslant I(x_j; y_j)\right\}.$$

By the definitions and notations of the probabilities $p_{j'}$ and $Q_{j',i'}$,

$$\begin{aligned} I(x_j;(z_{j-1},y_j)) &= H(z_{j-1},y_j) - H(z_{j-1},y_j|x_j) \\ &= H(z_{j-1}) + H(y_j|z_{j-1}) - H(z_{j-1},y_j|x_j) \\ &\overset{(a)}{=} H(z_{j-1}) + H(y_j|z_{j-1}) - H(z_{j-1}) \\ &= H(y_j|z_{j-1}) \\ &= \sum_{i=0}^{\ell} Pr(z_{j-1}=i) H(y_j|z_{j-1}=i) \\ &\overset{(b)}{=} \sum_{i=0}^{\ell-1} Pr(z_{j-1}=i) H(y_j|z_{j-1}=i) \\ &= \sum_{i=1}^{\ell-1} Q_{j-1,i} h\left(p_j\right) \\ &= \left(1 - Q_{j-1,\ell}\right) h\left(p_j\right). \end{aligned}$$

Step (a) follows from $H((z_{j-1}, y_j)|x_i) = H(z_{j-1}|x_j)$ since y_j is a function of x_j, z_{j-1}, and $H(z_{j-1}|x_j) = H(z_{j-1})$ because z_{j-1} is independent on x_j. Step (b) is implied by $H(y_j|z_{j-1} = \ell) = 0$. Hence, we can achieve the region $\widetilde{\mathcal{C}}_{t,\ell}$ for the ℓ-change t-write WℓM EIP:DIA model for the ϵ-error case.

The proof of the converse part is similar to the proof of this part in Theorem 2. Let S_j, $\hat{S}_j$, V_j, $1 \leqslant j \leqslant t$, and L be defined as in the proof of the converse part in Theorem 2. Thus, exactly as proved in Theorem 2, we have $I(X_j; Y_j|V_{j-1}) \geqslant I(S_j; \hat{S}_j|V_{j-1})$, $I(S_j; \hat{S}_j|V_{j-1}) \geqslant \log(M_j) - H(p_{e_j}) - p_{e_j}\log(M_j)$, and

$$\frac{1}{n} I(X_j; Y_j|V_{j-1}) \leqslant \sum_{i=0}^{\ell-1} Pr(V_{j-1,L} = i) H(Y_{j,L}|V_{j-1,L} = i).$$

Now, we set $p_j = Pr(X_{j,L} = 1)$. Thus, for $i < \ell$ $H(Y_{j,L}|V_{j-1,L} = i) = h(p_j)$. We also define for $i \in [\ell+1]$ $Q_{j,i} = Pr(V_{j,L} = i)$ and we note that $Q_{j,i}$ can be calculated as in Equation (4). Then

$$\begin{aligned} \frac{\log(M_j)}{n} - \epsilon_j &\leqslant \frac{1}{n} I(X_j; Y_j|V_{j-1}) \\ &\leqslant \sum\nolimits_{i=0}^{\ell-1} Pr(V_{j-1,L} = i) H(Y_{j,L}|V_{j-1,L} = i) \\ &= \sum\nolimits_{i=1}^{\ell-1} Q_{j-1,i} h\left(p_j\right) = \left(1 - Q_{j-1,\ell}\right) h\left(p_j\right), \end{aligned}$$

where $\epsilon_j = \frac{H(p_{e_j}) + p_{e_j}\log(M_j)}{n}$, and the theorem is implied. ■

Theorem 18: For $t > \ell \geqslant 2$, $\mathcal{R}_{t,\ell}^{EU:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon}$, and hence $\mathcal{C}_{t,\ell}^{EU:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$.

Proof: Let $\overline{\mathbf{R}} = (\overline{R}_1, \overline{R}_2, \ldots, \overline{R}_t)$ be a rate tuple which achieves the maximum sum-rate $\mathcal{R}_{t,\ell}^{EU:DIA,\epsilon}$, and we denote by $\overline{p}_j$ and $\overline{Q}_{j,i}$, $1 \leqslant j \leqslant t$ and $i \in [\ell+1]$, the probabilities which attain $\overline{\mathbf{R}}$ in $\overline{\mathcal{C}}_{t,\ell}$.

Now we present a rate tuple $\widetilde{\mathbf{R}} = (\widetilde{R}_1, \widetilde{R}_2, \ldots, \widetilde{R}_t) \in \widetilde{\mathcal{C}}_{t,\ell} > \overline{\mathbf{R}}$. Then, we conclude that $\widetilde{\mathbf{R}} \in \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon} \setminus \mathcal{C}_{t,\ell}^{EU:DIA,\epsilon}$, which implies that $\mathcal{R}_{t,\ell}^{EU:DIA,\epsilon} < \mathcal{R}_{t,\ell}^{EIP:DIA,\epsilon}$ and $\mathcal{C}_{t,\ell}^{EU:DIA,\epsilon} \subsetneq \mathcal{C}_{t,\ell}^{EIP:DIA,\epsilon}$.

We assume now that ℓ is even, while the proof for the odd case is similar. Since $\overline{\mathbf{R}}$ achieves maximum sum-rate we have $\overline{p}_t = 0.5$. For all j and i, $1 \leqslant j \leqslant t-2$ and $i \in [\ell]$, we define $\widetilde{p}_{j,0} = \widetilde{p}_{j,1} = \overline{p}_j$. In addition, $\widetilde{p}_{t-1,0} = 0.5$, $\widetilde{p}_{t-1,1} = \overline{p}_{t-1}$, and $\widetilde{p}_{t,0} = \widetilde{p}_{t,1} = 0.5$.

Thus, for all j and i, $1 \leqslant j \leqslant t-2$ and $i \in [\ell]$, $\widetilde{R}_j = \overline{R}_j$ and $\widetilde{Q}_{j,i} = \overline{Q}_{j,i}$. For the $(t-1)$-th write we have, $\widetilde{R}_{t-1} = \overline{Q}_{t-2,o} h(\overline{p}_{t-1}) + (\overline{Q}_{t-2,e} - \overline{Q}_{t-2,\ell})$ while $\overline{R}_{t-1} = (1 - \overline{Q}_{t-2,\ell}) h(\overline{p}_{t-1})$, and for the last write $\widetilde{R}_t = \overline{R}_t = 1 - \overline{Q}_{t-1,\ell}$,

Now we prove that $\overline{p}_{t-1} < 0.5$ which immediately implies that $\widetilde{R}_{t-1} > \overline{R}_{t-1}$ and thus completes the proof. Recall that $\overline{R}_t = 1 - \overline{Q}_{t-1,\ell} = 1 - \overline{Q}_{t-2,\ell} - \overline{Q}_{t-2,\ell-1}\overline{p}_{t-1}$. Thus, given the probabilities for the first $t-2$ writes, in order to achieve the maximal rate tuple $\overline{\mathbf{R}}$ we have to maximize $\overline{R}_{t-1} + \overline{R}_t$. That is, we choose $\overline{p}_{t-1}$ which maximizes $(1 - \overline{Q}_{t-2,\ell}) h(\overline{p}_{t-1}) - \overline{Q}_{t-2,\ell-1}\overline{p}_{t-1}$. The derivative is $(1 - \overline{Q}_{t-2,\ell}) \log(\frac{1-\overline{p}_{t-1,1}}{\overline{p}_{t-1,1}}) - \overline{Q}_{t-2,\ell-1}$, and the maximum is obtained for $\overline{p}_{t-1} = 1/(1 + 2^{\overline{Q}_{t-2,\ell-1}/(1-\overline{Q}_{t-2,\ell})})$. Since $\overline{\mathbf{R}}$ is maximal and $t > \ell \geqslant 2$, we have $\overline{Q}_{t-2,\ell-1} > 0$, and therefore $\overline{p}_{t-1} \neq 0.5$. ■

REFERENCES

[1] Y. M. Chee, T. Etzion, H. M. Kiah, and A. Vardy, "Cooling codes: Thermal-management coding for high-performance interconnects," *IEEE Trans. Inf. Theory*, vol. 64, no. 4, pp. 3062–3085, Apr. 2018.

[2] Y. M. Chee, H. M. Kiah, A. Vardy, and E. Yaakobi, "Explicit constructions of finite-length WOM codes," *IEEE Trans. Inf. Theory*, vol. 66, no. 5, pp. 2669–2682, May 2020.

[3] Y. M. Chee, H. M. Kiah, A. J. Han Vinck, V. K. Vu, and E. Yaakobi, "Coding for write ℓ-step-up memories," in *Proc. IEEE Int. Symp. Inform. Theory*, Jul. 2019, pp. 1597–1601.

[4] Y. M. Chee, M. Horovitz, A. Vardy, V. K. Vu, and E. Yaakobi, "Codes for endurance-limited memories," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Singapore, Oct. 2018, pp. 501–505.

[5] Y. M. Chee, M. Horovitz, A. Vardy, V. K. Vu, and E. Yaakobi, "Endurance-limited memories with informed decoder," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Visby, Sweden, Aug. 2019, pp. 1–5.

[6] Y.-S. Chen *et al.*, "Robust high-resistance state and improved endurance of HfO_X resistive memory by suppression of current overshoot," *IEEE Electron Device Lett.*, vol. 32, no. 11, pp. 1585–1587, Nov. 2011.

[7] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, NJ, USA: Wiley, 2012.

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

[9] A. Grossi *et al.*, "Resistive RAM endurance: Array-level characterization and correction techniques targeting deep learning applications," *IEEE Trans. Electron Devices*, vol. 66, no. 3, pp. 1281–1288, Mar. 2019.

[10] C. Heegard, "On the capacity of permanent memory," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 1, pp. 34–42, Jan. 1985.

[11] M. Horovitz and E. Yaakobi, "On the capacity of write-once memories," *IEEE Trans. Inf. Theory*, vol. 63, no. 8, pp. 5124–5137, Aug. 2017.

[12] Y. Kim *et al.*, "Locally rewritable codes for resistive memories," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2470–2485, Sep. 2016.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.

[13] T. Kobayashi, H. Morita, and A. Manada, "On the capacity of write-constrained memories," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 5101–5109, Jul. 2018.
[14] R. Maddah, R. Melhem, and S. Cho, "RDIS: Tolerating many stuck-at faults in resistive memory," *IEEE Trans. Comput.*, vol. 64, no. 3, pp. 847–861, Mar. 2015.
[15] C. D. Nguyen, V. K. Vu, and K. Cai, "Two-dimensional weight-constrained codes for crossbar resistive memory arrays," *IEEE Commun. Lett.*, vol. 25, no. 5, pp. 1435–1438, May 2020.
[16] A. M. Rana *et al.*, "Endurance and cycle-to-cycle uniformity improvement in tri-layered CeO_2/Ti/CeO_2 resistive switching devices by changing top electrode material," *Sci. Rep.*, vol. 7, no. 1, p. 39539, Jan. 2017.
[17] R. L. Rivest and A. Shamir, "How to reuse a 'write-once' memory," *Inf. Control*, vol. 55, nos. 1–3, pp. 1–19, Oct. 1982.
[18] G. Sassine *et al.*, "Sub-pJ consumption and short latency time in RRAM arrays for high endurance applications," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, Mar. 2018, pp. 1–5.
[19] S. Schechter, G. H. Loh, K. Straus, and D. Burger, "Use ECP, not ECC, for hard failures in resistive memories," in *Proc. 37th Annu. Int. Symp. Comput. Archit. (ISCA)*, 2010, pp. 141–152.
[20] A. Shpilka, "New constructions of WOM codes using the Wozencraft ensemble," *IEEE Trans. Inf. Theory*, vol. 59, no. 7, pp. 4520–4529, Jul. 2013.
[21] A. Shpilka, "Capacity-achieving multiwrite WOM codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 3, pp. 1481–1487, Mar. 2014.
[22] G. Wang *et al.*, "Improving resistance uniformity and endurance of resistive switching memory by accurately controlling the stress time of pulse program operation," *Appl. Phys. Lett.*, vol. 106, no. 9, Mar. 2015, Art. no. 092103.
[23] J. K. Wolf, A. D. Wyner, J. Ziv, and J. Korner, "Coding for a write-once memory," *AT&T Bell Labs. Tech. J.*, vol. 63, no. 6, pp. 1089–1112, Aug. 1984.
[24] C. Xu, D. Niu, Y. Zheng, S. Yu, and Y. Xie, "Impact of cell failure on reliable cross-point resistive memory design," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 20, no. 4, pp. 63:1–63:21, Sep. 2015.
[25] F.-Y. Yuan *et al.*, "Conduction mechanism and improved endurance in HfO_2-based RRAM with nitridation treatment," *Nanosc. Res. Lett.*, vol. 12, no. 1, p. 574, Oct. 2017.
[26] F. Zahoor, T. Z. A. Zulkifli, and F. A. Khanday, "Resistive random access memory (RRAM): An overview of materials, switching mechanism, performance, multilevel cell (MLC) storage, modeling, and applications," *Nanosc. Res. Lett.*, vol. 15, no. 1, pp. 1–26, Apr. 2020.
[27] L. Zhang, B. Neely, D. Franklin, D. Strukov, Y. Xie, and F. T. Chong, "Mellow writes: Extending lifetime in resistive memories through selective slow write backs," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 519–531.
[28] M. Zhao *et al.*, "Characterizing endurance degradation of incremental switching in analog RRAM for neuromorphic systems," in *IEDM Tech. Dig.*, Dec. 2018, pp. 468–471.

Yeow Meng Chee (Senior Member, IEEE) received the B.Math. degree in computer science and combinatorics and optimization and the M.Math. and Ph.D. degrees in computer science from the University of Waterloo, Waterloo, ON, Canada, in 1988, 1989, and 1996, respectively. He was a Professor with the School of Physical and Mathematical Sciences and the Interim Dean of science with Nanyang Technological University. He is currently a Professor with the Department of Industrial Systems Engineering and Management and the Associate Vice President of innovation and enterprise with the National University of Singapore. His research interests include the interplay between combinatorics and computer science/engineering, particularly combinatorial design theory, coding theory, extremal set systems, and electronic design automation.

Michal Horovitz (Member, IEEE) received the B.Sc. degree from the Department of Mathematics and the Department Computer Science, Open University, Ra'anana, Israel, in 2009, and the Ph.D. degree from the Computer Science Department, Technion—Israel Institute of Technology, Haifa, Israel, in 2017. She is currently a Lecturer with the Department of Computer Science, Tel-Hai College, Israel. She is also a Researcher at the Galilee Research Institute—Migal, Upper Galilee, Israel. Her research interests include coding theory with applications to non-volatile memories and DNA storage, information theory, combinatorics, machine learning, and optimization.

Alexander Vardy (Fellow, IEEE) was born in Moscow, Russia, in 1963. He received the B.Sc. degree *(summa cum laude)* from the Technion—Israel Institute of Technology, Israel, in 1985, and the Ph.D. degree from Tel-Aviv University, Israel, in 1991.

From 1985 to 1990, he was with the Israeli Air Force, where he worked on electronic counter measures systems and algorithms. From 1992 to 1993, he was a Visiting Scientist with the IBM Almaden Research Center, San Jose, CA, USA. From 1993 to 1998, he was with the University of Illinois at Urbana-Champaign, first as an Assistant Professor and then as an Associate Professor. Since 1998, he has been with the University of California at San Diego (UCSD), where he is currently the Jack Keil Wolf Chair Professor with the Department of Electrical and Computer Engineering and the Department of Computer Science. While on sabbatical from UCSD, he has held long-term visiting appointments with CNRS, France, the EPFL, Switzerland, the Technion—Israel Institute of Technology, and Nanyang Technological University, Singapore. His research interests include error-correcting codes, algebraic and iterative decoding algorithms, lattices and sphere packings, coding for storage systems, cryptography, computational complexity theory, and fun math problems. He has been a member of the Board of Governors of the IEEE Information Theory Society from 1998 to 2006 and from 2011 to 2017. In 1996, he became a fellow of the David and Lucile Packard Foundation. He received the IBM Invention Achievement Award in 1993 and the NSF Research Initiation and CAREER Awards in 1994 and 1995, respectively. In 1996, he was appointed as a fellow of the Center for Advanced Study, University of Illinois, and received the Xerox Award for Faculty Research. He received the IEEE Information Theory Society Paper Award (jointly with Ralf Koetter) in 2004. In 2005, he received the Fulbright Senior Scholar Fellowship and the Best Paper Award at the IEEE Symposium on Foundations of Computer Science (FOCS). In 2017, his work on polar codes was recognized by the IEEE Communications and Information Theory Societies Joint Paper Award. From 1995 to 1998, he was an Associate Editor for *Coding Theory*. From 1998 to 2001, he was the Editor-in-Chief of the IEEE TRANSACTIONS ON INFORMATION THEORY.

Van Khu Vu received the B.Sc. degree in mathematics from Vietnam National University (VNU), Hanoi, in 2010, and the Ph.D. degree in mathematics from Nanyang Technological University (NTU), Singapore, in 2018. From 2010 to 2012, he was a Lecturer at the VNU College of Sciences, Hanoi. From 2018 to 2019, he was a Research Fellow at the School of Physical and Mathematical Sciences, NTU, Singapore. He is currently a Research Fellow at the Department of Industrial Systems Engineering and Management, National University of Singapore. His primary research interests lie in the areas of algorithms, combinatorics, and coding theory.

Eitan Yaakobi (Senior Member, IEEE) received the B.A. degree in computer science and mathematics and the M.Sc. degree in computer science from the Technion—Israel Institute of Technology, Haifa, Israel, in 2005 and 2007, respectively, and the Ph.D. degree in electrical engineering from the University of California at San Diego in 2011. From 2011 to 2013, he was a Post-Doctoral Researcher with the Department of Electrical Engineering, California Institute of Technology, and the Center for Memory and Recording Research, University of California at San Diego. He is currently an Associate Professor at the Computer Science Department, Technion—Israel Institute of Technology. His research interests include information and coding theory with applications to non-volatile memories, associative memories, DNA storage, data storage and retrieval, and private information retrieval. He received the Marconi Society Young Scholar in 2009 and the Intel Ph.D. Fellowship in 2010–2011.

Authorized licensed use limited to: Technion Israel Institute of Technology. Downloaded on February 23,2022 at 17:36:38 UTC from IEEE Xplore. Restrictions apply.