Fast-Readable Share Codes for Flash Memory

Roee Gross*, **Roni Con*** and **Eitan Yaakobi***
*Department of Computer Science, Technion—Israel Institute of Technology, Haifa 3200003, Israel
Email: {roee.g, roni.con, yaakobi}@cs.technion.ac.il

***Abstract*—Multi-level flash memory increases storage density by allowing each cell to store multiple voltage levels, but this often increases read latency due to repeated threshold sensing. *SHare Codes* (*SHC*), introduced recently by Shibata et al. in [1], provide a flexible coding framework that enables fast read access by distributing information across cells and allowing cell-specific threshold tests. We continue the study of SHC schemes that operate at full storage capacity, with the goal of minimizing the average read cost required to retrieve the stored data. We derive general upper and lower bounds and present explicit constructions with improved performance compared to simple Gray-code-based schemes.**

I. Introduction

The rapid proliferation of digital applications from mobile devices to hyperscale data centers serving AI workloads has driven a sustained increase in demand for flash memory. Meeting this demand requires not only higher-capacity storage but also *fast read access*. A standard approach to increasing storage density is multi-level cell (MLC) technology, which stores multiple bits in each cell. However, MLC typically increases read latency, particularly in the number of threshold-voltage tests required to retrieve information.

While read latency is a major bottleneck in MLC flash, a significant body of work has focused on optimizing the write process. In particular, write-once memory (WOM) codes exploit the asymmetric nature of flash cells where voltage levels can only be increased between erase operations to enable multiple logical writes to the same physical cells [2]–[15]. These schemes improve endurance and reduce erase frequency, but do not address the cost of reading information from multi-level cells.

To address this challenge, various coding schemes have been proposed to optimize the trade-off between storage density and read performance. One such approach is *Random Input–Output (RIO)* codes [16] [17], which enable reading a bit using the same single threshold test across all cells, but sacrifice storage capacity. A more general framework, introduced in [1], [18], [19], is the novel *SHare Code (SHC)* paradigm, in which information is distributed across multiple cells, allowing parallel single-threshold operations. This scheme achieves faster reads while preserving the structure of multi-level storage.

An $SHC(n,k)$ system consists of n memory cells, where each cell can store one of 2^k possible voltage levels. Overall, the system stores a binary vector of length nk. To recover the stored data, the reader performs *threshold operations* on the memory cells. Intuitively, a threshold operation checks whether the voltage of a cell exceeds a certain level. Unlike RIO codes, which use the same set of thresholds for all cells, SHC schemes allow each cell to be queried using its own (possibly different) set of thresholds. As shown in [1], [18], [19], this added flexibility can significantly reduce the average number of threshold operations required for reading.

More specifically, assume that $\boldsymbol{v}$ is the stored vector in n cells and v_i is the bit we want to retrieve. Then, on every cell $j \in [n]$, we perform t_j threshold operations and collect the outputs to decode the i-th bit of the input. The cost of reading the i-th bit is the maximum number of threshold operations performed on any single cell and we define the total reading cost of the scheme as the average cost across input bits. This is the central parameter of the scheme, and we aim to minimize it.

II. Problem Definition and Our Results

For a positive integer n, we denote by $[n] = \{1, \ldots, n\}$ the set of all natural numbers between 1 and n. For a set S, we denote its cardinality by $|S|$ and its power set by $\mathcal{P}(S)$. Lowercase bold letters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ denote binary vectors and the zero vector of length n is denoted by $\mathbf{0}^n$. Furthermore, addition of binary bits or binary vectors is performed over the field $\mathbb{F}_2$.

Our definition of an SHC scheme is more general than that given in [1]. We choose to present it in order to point out that one can consider studying extensions of the model in order to achieve better performance. We begin by giving the formal definition of a threshold operation and then we define an SHC scheme.

Definition II.1 (Threshold operation)**.** *Let $a \in [\ell]$. A threshold operation with respect to a is denoted as $\tau_a : [\ell] \to \{0,1\}$ and is defined by $\tau_a(c) = 1$ if $c \geq a$ and $\tau_a(c) = 0$ otherwise.*

Definition II.2 (SHC)**.** *An $\mathrm{SHC}(n, \ell, b)$ scheme is an encoding scheme consisting of n cells, where each cell can store one of ℓ distinct voltage levels. The scheme represents a total of b information bits and is defined by an encoding function $E : \{0,1\}^b \to [\ell]^n$, which maps the b information bits to a voltage level for each of the n cells, we denote $(c_1, \ldots, c_n) = E(\boldsymbol{v})$, and by b decoding tuples $(R_1, G_1, D_1), \ldots, (R_b, G_b, D_b)$ where for each bit $i \in [b]$*

- *$R_i : [n] \to \mathcal{P}([\ell])$ represents the threshold values to be applied to every cell.*
- *$G_i := (G_{i,1}, \ldots, G_{i,n})$ consists of n functions that take as input the threshold operation outputs and output an element in a finite alphabet Σ. Formally, for all $j \in [n]$ $G_{i,j} : \{0,1\}^{|R_i(j)|} \to \Sigma$, and it gets as input $(\tau_r(c_j) : r \in R_i(j))$.*
- *$D_i : \Sigma^n \to \mathbb{F}_2$ is the decoding function that gets as input the outputs of the $G_{i,j}$'s and outputs a bit.*

The encoding map E and the decoding tuples $(R_1, G_1, D_1), \ldots, (R_b, G_b, D_b)$ need to satisfy the

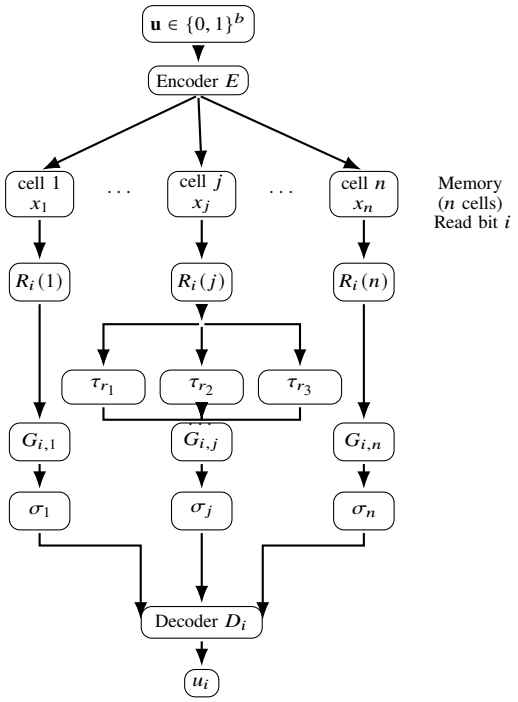

Fig. 1: Encoding and single-bit readout in an n-cell memory scheme.

following. For every $\mathbf{v} \in \{0,1\}^b$, and for every $i \in [b]$, we have that $D_i(\sigma_1, \ldots, \sigma_n) = v_i$, where for all $j \in [n]$,$\sigma_j = G_{i,j}\left(\tau_r(c_j) : r \in R_i(j)\right)$.

An illustration of an SHC scheme is depicted in Fig. 1. As discussed in the introduction, the main parameter one wants to optimize is the number of threshold operations that are needed in order to read the data. The formal definition of the *average read cost* of an SHC scheme is given next.

Definition II.3. *Given an* $\mathrm{SHC}(n, \ell, b)$ *scheme which is denoted by* $\mathcal{S}$*, its* reading cost *of bit i is defined as* $\mathrm{RC}_i(\mathcal{S}) \triangleq \max_{j\in[n]} |R_i(j)|$. *Its* average reading cost *is* $\mathrm{RC}(\mathcal{S}) \triangleq \frac{1}{b}\sum_{i=1}^{b} \max_{j\in[n]} |R_i(j)|$.

Given the above definitions, the goal is to design an SHC scheme with as small as possible average read cost.

Remark II.1. *We point out that the functions $G_{i,j}$ in Definition II.2 can be limited by the hardware itself. For example, in [18], the authors considered only the case in which every $G_{i,j}$ outputs a single bit, regardless of the number of thresholds that were performed. In this paper, we will follow this setting as well since it is simpler to implement and aligns with standard read mechanisms such as RIO codes, discussed in the introduction.*

However, from an information-theoretic perspective, outputting a single bit from each cell, regardless of the number of thresholds applied, discards potentially valuable information. A more refined approach would be to output a range, providing a finer description of the stored voltage level and potentially reducing the average read cost. We leave this for future work.

In this paper, we will focus on the following definition of a XOR-SHC scheme. We will work in the *full capacity* regime and also assume that $\ell = 2^k$ is a power of 2, that is, we will assume that $b = n \cdot \log_2 \ell = n \cdot k$.

Definition II.4 (XOR-SHC, [18])**.** *Let n and k be positive integers. An* $\mathrm{SHC}(n, 2^k, nk)$ *scheme is called an* $\text{XOR-SHC}(n, k)$ *scheme if in Definition II.2 we set $\Sigma = \{0,1\}$ and for all $i \in [nk], j \in [n]$ we have $G_{i,j}\left(\tau_r(c_j) : r \in R_i(j)\right) = \sum_{r\in R_i(j)} \tau_r(c_j)$.*

We will write $\mathcal{S} \in \text{XOR-SHC}(n, k)$ when $\mathcal{S}$ is an XOR-SHC scheme with these parameters.

Example II.1. *Consider an $\mathcal{S} \in \text{XOR-SHC}(2,2)$ scheme defined by the encoding function $E : \mathbb{F}_2^4 \to [4]^2$ with*

$\boldsymbol{v}$	$E(\boldsymbol{v})$	$\boldsymbol{v}$	$E(\boldsymbol{v})$
0000	*(1,1)*	*1000*	*(2,1)*
0001	*(4,2)*	*1001*	*(3,2)*
0010	*(1,2)*	*1010*	*(2,2)*
0011	*(4,1)*	*1011*	*(3,1)*
0100	*(4,4)*	*1100*	*(3,4)*
0101	*(4,3)*	*1101*	*(3,3)*
0110	*(1,3)*	*1110*	*(2,3)*
0111	*(1,4)*	*1111*	*(2,4)*

For each bit $i \in \{1,2,3,4\}$, the threshold sets R_i are:

$$\begin{aligned} R_1(1) &= \{2,4\}, & R_1(2) &= \varnothing, \\ R_2(1) &= \varnothing, & R_2(2) &= \{3\}, \\ R_3(1) &= \{3\}, & R_3(2) &= \{2\}, \\ R_4(1) &= \{3\}, & R_4(2) &= \{4\}. \end{aligned}$$

and for all $i \in [4]$, we define $D_i(x,y) = x + y$. Therefore, the reading costs are

$$\mathrm{RC}_1(\mathcal{S}) = 2, \quad \mathrm{RC}_2(\mathcal{S}) = 1, \quad \mathrm{RC}_3(\mathcal{S}) = 1, \quad \mathrm{RC}_4(\mathcal{S}) = 1.$$

Hence, the average reading cost is $\mathrm{RC}(\mathcal{S}) = 5/4$.

Definition II.5. *We define* $\mathrm{RC}^*(n,k) = \min_{\mathcal{S}\in\text{XOR-SHC}(n,k)} \mathrm{RC}(\mathcal{S})$*, to be the minimum average reading cost achievable by any scheme in* $\text{XOR-SHC}(n,k)$.

For two-cell schemes with small parameters, an exhaustive search in [18] established that $\mathrm{RC}^*(2,2) = \frac{5}{4}$, $\mathrm{RC}^*(2,3) = \frac{10}{6}$, and $\mathrm{RC}^*(2,4) \le \frac{22}{8}$. Moreover, for three-cell schemes, it was shown that $\mathrm{RC}^*(3,2) = 1$.

Our main result in the paper is a general construction for XOR-SHC schemes with two cells, formulated in the following theorem. However, the method we employ can be naturally extended to schemes with a larger number of cells.

Theorem II.1. *For every integer $n \ge 1$ and every integer $k \ge 3$, it holds that*

$$\mathrm{RC}^*(n,k) \le \begin{cases} \dfrac{(3/4)\,2^k - 1}{k}, & \text{if } n \text{ is even}, \\ \dfrac{\left((3/4) + \frac{1}{4n}\right)2^k - 1}{k}, & \text{if } n \text{ is odd}. \end{cases}$$

Moreover, we obtain better upper bounds than Theorem II.1 for the specific cases of $\mathrm{RC}^*(2,5)$ and $\mathrm{RC}^*(2,6)$;

TABLE I: Comparison of lower bounds, achieved values (including this work), and Gray-code performance for $n = 2$. All values are average reading costs.

k	**Lower Bound**	**Achieved Value**	**Gray Code**
2	—	1.25 (tight) [18]	1.5
3	—	10/6 (tight) [18]	7/3
4	1.875	2.75 [18]	3.75
5	3.1	4.5 (this work)	6.2
6	5.25	7.5 (this work)	10.5
$k \geq 3$	$\frac{2^k-1}{2k}$	$\frac{3/4\cdot 2^k-1}{k}$ (this work)	$\frac{2^k-1}{k}$

see Table I for the exact numbers and comparison with [18] (The table includes comparisons with a Gray code-based scheme, which is introduced in the following section).

III. Elementary Bounds

In this section we will present simple lower and upper bounds on $\mathrm{RC}^*(n,k)$. We start by using a Gray code to construct an XOR-SHC$(1,k)$ which would immediately imply an upper bound on $\mathrm{RC}^*(1,k)$.

Claim III.1. *We have that* $\mathrm{RC}^*(1,k) \leq \frac{2^k-1}{k}$.

Proof. Fix a natural number k, and consider a *binary Gray code* of length k, that is, a bijection $G : [2^k] \to \mathbb{F}_2^k$, such that for all $j \in [2^k-1]$, the Hamming distance between $G(j)$ and $G(j+1)$ is exactly 1. We also assume that $G(1) = \mathbf{0}^k$.

We now define a XOR-SHC$(1,k)$ scheme as follows. Let the encoding map $E : \{0,1\}^k \to [2^k]$ be the inverse map of the Gray code G. The threshold set for the i-th bit is defined to be

$$R_1(i) = \left\{ j \in [2^k] \mid j \geq 2, G(j-1) + G(j) = e_i \right\},$$

namely the set of all indices at which the i-th bit flips as the Gray code advances from position $j-1$ to j. Finally, we define $D : \{0,1\} \to \{0,1\}$ to be the identity function.

We now prove the correctness of this scheme and compute its reading cost. Let $\boldsymbol{v} \in \{0,1\}^k$ be the data we want to store, and assume we want to recover the i-th bit of $\boldsymbol{v}$ from the stored voltage value $c = E(\boldsymbol{v})$. Since E is the inverse of a Gray code G, we can determine the value v_i by counting the number of times the i-th bit flips in the following sequence of vectors $G(0), G(1), \ldots, G(c)$. This information is given by applying the thresholds indicated in $R_1(i)$. The claim about the average reading cost follows by noting that since G is a bijection, we have $\sum_{i=1}^{k} |R_1(i)| = 2^k - 1$. □

Remark III.1. *Observe that the claim concerns the* average *read cost. For this purpose, the only property required of the code is that it be a Gray code, namely, that exactly one bit changes between any two consecutive codewords. One may further employ a* balanced *Gray code, in which each bit flips approximately the same number of times throughout the sequence [20]–[22]. This ensures that the read cost of the most expensive bit is close to the average read cost.*

Lemma III.1. *For all* $n, k \in \mathbb{N}$*, we have* $\mathrm{RC}^*(n,k) \geq \frac{2^k-1}{nk}$.

Proof. Assume that $\mathrm{RC}^*(n,k) < \frac{2^k-1}{nk}$. This implies that there exists a scheme with $\sum_{i=1}^{nk} \max_{j\in[n]} |R_i(j)| < 2^k - 1$. In particular, $\sum_{i=1}^{nk} |R_i(1)| < 2^k - 1$. Recall that any storage cell can store 2^k storage values. Thus, there exists $r \in [2^k - 1]$ such that for all $a \in \cup_{i\in[nk]} R_i(1)$, we have that $\tau_a(r) = \tau_a(r+1)$. Namely, all thresholds that can be applied on the first memory cell will return the same value on the stored voltages r and $r+1$.

Since the encoding map $E : \{0,1\}^{nk} \to [2^k]^n$ is a bijection, there are two distinct $\boldsymbol{v}, \boldsymbol{u} \in \{0,1\}^{nk}$ that are encoded to $(r, 1, \ldots, 1)$ and $(r+1, 1, \ldots, 1)$, respectively. On both, the decoding of every bit $i \in [nk]$ will return the same value, contradicting the correctness of the scheme. □

As a corollary of Claim III.1 and Lemma III.1, we have that for $n = 1$, $\mathrm{RC}^*(1,k) = (2^k - 1)/k$.

Observe that if we are given two schemes SHC(n_1, k) and SHC(n_2, k), they can be concatenated into a single scheme of type SHC(n_1+n_2, k). This is formalized in the following lemma.

Lemma III.2. *Let* $k, n_1, n_2 \in \mathbb{N}$*, such that* $n_1, n_2 \geq 1$*, then*

$$\mathrm{RC}^*(n_1+n_2,\ k) \leq \frac{n_1}{n_1+n_2}\mathrm{RC}^*(n_1,k) + \frac{n_2}{n_1+n_2}\mathrm{RC}^*(n_2,k).$$

Proof. Let $\mathcal{S}_i$ be an XOR-SHC(n_i, k) scheme that attains the minimal average reading cost $\mathrm{RC}^*(n_i,k)$ for $i \in [2]$. Construct a scheme $\mathcal{S} \in$ XOR-SHC(n_1+n_2, k) as follows. By using n_1 cells and the scheme $\mathcal{S}_1$, we can store $n_1 k$ bits and recover them with an average read cost of $\mathrm{RC}^*(n_1,k)$. Using the additional n_2 cells and the scheme $\mathcal{S}_2$, we can store $n_2 k$ bits with an average read cost of $\mathrm{RC}^*(n_2,k)$. Thus, the average reading cost of $\mathcal{S}$ is simply

$$\mathrm{RC}(\mathcal{S}) = \frac{n_1}{n_1+n_2}\mathrm{RC}^*(n_1,k) + \frac{n_2}{n_1+n_2}\mathrm{RC}^*(n_2,k).$$

The lemma follows from $\mathrm{RC}^*(n_1+n_2,k) \leq \mathrm{RC}(\mathcal{S})$. □

Corollary III.1. *For all* $n \in \mathbb{N}$*, we have* $\mathrm{RC}^*(n,k) \leq \frac{2^k-1}{k}$.

Proof. Let S be the XOR-SHC$(1,k)$ scheme constructed in Claim III.1. Then, the corollary follows by simply applying Lemma III.2. □

As an immediate corollary, we obtain

Corollary III.2. $\frac{2^k-1}{nk} \leq \mathrm{RC}^*(n,k) \leq \frac{2^k-1}{k}$.

IV. Beyond the Trivial Bound

The number of possible encoding schemes is a function of n and k, and thus it is tempting to attempt an exhaustive search in order to identify an optimal scheme. This was performed in [18] for small values of n and k. However, very quickly, the number of possibilities becomes prohibitively large.

In this section we present an efficient and explicit construction of XOR-SHC schemes with an improved upper bound over the simple upper bound provided in the previous section. To this end, we define a different object, called a *complete vector system* which consists of vector sequences that satisfy a property. Then, we will describe how to construct an XOR-SHC scheme from a complete vector system. Finally, we will present an efficient construction of complete vector systems.

We start by introducing some new notations. Uppercase letters such as A, B, C, in this section will denote sequences

of vectors. Formally, $A \in (\mathbb{F}_2^n)^k$ denotes a sequence of length k consisting of length-n binary vectors. For a sequence of vectors A, we denote by A_i its i-th vector and $(A_i)_j$ refers to the j-th coordinate of the i-th vector in the sequence A.

Definition IV.1 (Complete vector system)**.** *Let $n, k \in \mathbb{N}$. A* complete vector system *with parameters n and k is a collection of sequences $\mathcal{A} = (A_1, \ldots, A_n)$ where each A_i is a sequence of 2^k binary vectors of length nk, such that the function $F : [2^k]^n \to \mathbb{F}_2^{nk}$, $\quad F(x_1, \ldots, x_n) = \sum_{i=1}^{n} (A_i)_{x_i}$, is bijective, and $(A_i)_1 = \mathbf{0}$ for each $i \in [n]$. We write $\mathcal{A} \in \mathrm{CVS}(n, k)$ to say that $\mathcal{A}$ is a complete vector system with parameters n and k.*

Example IV.1. *$\mathcal{A} = (A_1, A_2) \in \mathrm{CVS}(2, 3)$, where*

$$\begin{aligned} A_1 = (&000000,\ 100000,\ 111000,\ 011000,\\ &011010,\ 011110,\ 000110,\ 000010),\\ A_2 = (&000000,\ 010000,\ 110100,\ 100100,\\ &100101,\ 101101,\ 001001,\ 000001). \end{aligned}$$

We now define the cost of a complete vector system.

Definition IV.2. *Let $\mathcal{A} = (A_1, \ldots, A_n) \in \mathrm{CVS}(n, k)$. For each index $j \in [n]$ and bit position $i \in [nk]$, define*

$$\mathrm{cost}_{\mathcal{A}}(i, j) = \sum_{\ell=2}^{2^k} ((A_j)_\ell)_i - ((A_j)_{\ell-1})_i \ .$$

Intuitively, the cost counts the number of times the i-th bit flips in the j-th sequence. The total cost associated with bit *$i \in [nk]$ is then defined as $\mathrm{TC}_i(\mathcal{A}) = \max_{j \in [n]} \mathrm{cost}_{\mathcal{A}}(i, j)$, and the* total cost of *$\mathcal{A}$ is $\mathrm{TC}(\mathcal{A}) = \sum_{i \in [nk]} \mathrm{TC}_i(\mathcal{A})$.*

The following theorem establishes the connection between CVSs and XOR-SHC, as well as the correspondence between their respective cost measures.

Theorem IV.1. *Let $\mathcal{A} = (A_1, \ldots, A_n) \in \mathrm{CVS}(n, k)$. Then, there exists an $\mathcal{S} \in \text{XOR-SHC}(n, k)$ such that for every bit position $i \in [nk]$, $\mathrm{RC}_i(\mathcal{S}) = \mathrm{TC}_i(\mathcal{A})$.*

Proof. We define the corresponding XOR-SHC(n, k) scheme as follows. Let $F : [2^k]^n \to \mathbb{F}_2^{nk}$ denote the bijective mapping of $\mathcal{A}$ (see Definition IV.1) and let E, the encoding map of $\mathcal{S}$ be the inverse of F. Namely, $E(\boldsymbol{a}) = F^{-1}(\boldsymbol{a})$ computes the unique values $x_1, \ldots, x_n$ such that $F(x_1, \ldots, x_n) = \boldsymbol{a}$.

We now describe how to decode the i-th bit in the constructed XOR-SHC(n, k) and compute its reading cost. For each cell $j \in [n]$, we define $R_i(j) = \{\ell \in [2^k] \mid \ell \geq 2, ((A_j)_\ell)_i - ((A_j)_{\ell-1})_i = 1\}$.

Recall that by the definition of XOR-SHC (Definition II.4), the $G_{i,j}$s xors all the thresholds that were performed on the j-th cell. Finally, $D_i : \{0, 1\}^n \to \{0, 1\}$, is given by $D_i(\sigma_1, \ldots, \sigma_n) = \sum_{i=1}^{n} \sigma_i$.

We now verify that the decoding works. Let $\boldsymbol{v} \in \{0, 1\}^{nk}$ and let $i \in [nk]$. By our construction, the stored values $c_1, \ldots, c_n$ satisfy that $\sum_{j=1}^{n} (A_j)_{c_j} = \boldsymbol{v}$ and in particular,

$$\sum_{j=1}^{n} \left((A_j)_{c_j}\right)_i = v_i \ . \tag{1}$$

Now, observe that to determine $\left((A_j)_{c_j}\right)_i$, it is enough to count the number of times the i-th bit flips in the sequence $(A_j)_1, (A_j)_2 \ldots, (A_j)_{c_j}$. This is done by applying all the thresholds in $R_i(j)$. Indeed, by definition, $R_i(j)$ contains all $\ell \in [2^k]$, *such that* $\ell \geq 2$ for which the i-th bit $(A_j)_\ell$ and $(A_j)_{\ell-1}$ differ, and thus, performing threshold operations of these thresholds on the stored value c_j, exactly counts the number of times the i-th bit flipped. Thus, by performing these threshold operations on all cells, we recover v_i from (1).

The claim about the reading cost follows by noting that for all $i \in [nk]$ and $j \in [n]$, we have $\mathrm{RC}_i(S) = \max_{j \in [n]} |R_i(j)| = \max_{j \in [n]} \mathrm{cost}_{\mathcal{A}}(i, j) = \mathrm{RC}_i(\mathcal{A})$. □

Example IV.2. *For the following sequences,*

$$\begin{aligned} A_1 &= (0000, 1000, 1011, 0011),\\ A_2 &= (0000, 0010, 0110, 0111), \end{aligned}$$

the construction described above yields exactly the XOR-SHC *presented in Example II.1.*

We saw in Lemma III.2 that multiple SHCs can be concatenated into a single SHC. We now prove an analogous statement for complete vector systems.

Lemma IV.1. *Let $\mathcal{A} \in \mathrm{CVS}(n, k)$ and $\mathcal{B} \in \mathrm{CVS}(m, k)$. Then, there exists $C = (C_1, \ldots, C_{n+m}) \in \mathrm{CVS}(n + m, k)$ where $\mathrm{TC}(C) = \mathrm{TC}(\mathcal{A}) + \mathrm{TC}(\mathcal{B})$.*

Proof. Define $C = (C_1, \ldots, C_{n+m})$ by

$$C_j = \begin{cases} A_j \circ \mathbf{0}^{mk}, & j \in [n], \\ \mathbf{0}^{nk} \circ B_{j-n}, & j \in \{n+1, \ldots, n+m\} \ , \end{cases}$$

where $\boldsymbol{a} \circ A = (a \circ A_1, a \circ A_2, \ldots, a \circ A_{|A|})$ and $A \circ \boldsymbol{a}$ is defined analogously, where a is concatenated from the right, and the mapping $F_C : [2^k]^{n+m} \to \mathbb{F}_2^{(n+m)k}$ by $F_C(x_1, \ldots, x_{n+m}) = \sum_{i=1}^{n+m} (C_i)_{x_i}$. According to Definition IV.1, to show that $C \in \mathrm{CVS}(n + m, k)$, we need to prove that F_C is injective. First, observe that $F_C(x_1, \ldots, x_{n+m}) = \sum_{j=1}^{n+m} (C_j)_{x_j} = \sum_{j=1}^{n} (A_j)_{x_j} \circ \mathbf{0}^{mk} + \sum_{i=n+1}^{n+m} \mathbf{0}^{nk} \circ (B_i)_{x_i} = \left(\sum_{j=1}^{n} (A_j)_{x_j}\right) \circ \left(\sum_{i=n+1}^{n+m} (B_i)_{x_i}\right) = F_{\mathcal{A}}(x_1, \ldots, x_n) \circ F_{\mathcal{B}}(x_{n+1}, \ldots, x_{n+m})$, where $F_{\mathcal{A}}, F_{\mathcal{B}}$ are the bijective maps given by $\mathcal{A}$ and $\mathcal{B}$, respectively. It follows that F_C is also a bijection. We now compute the cost of C, $\mathrm{TC}(C) = \sum_{i=1}^{(n+m)k} \max_{j \in [n+m]} \mathrm{cost}_C(i, j) = \sum_{i=1}^{nk} \max_{j \in [n]} \mathrm{cost}_C(i, j) + \sum_{i=1}^{mk} \max_{j \in [m]} \mathrm{cost}_C(nk + i, n + j) = \sum_{i=1}^{nk} \max_{j \in [n]} \mathrm{cost}_{\mathcal{A}}(i, j) + \sum_{i=1}^{mk} \max_{j \in [m]} \mathrm{cost}_{\mathcal{B}}(i, j) = \mathrm{TC}(\mathcal{A}) + \mathrm{TC}(\mathcal{B})$, where the second equality follows since for every vector in C_j, if $j \in [n]$, then all of its last mk are zero and if $j > n$, all of its first nk are zero. □

Corollary IV.1. *Let $\mathcal{A} \in \mathrm{CVS}(n, k)$ and let $\alpha \in \mathbb{N}$. Then, there exists $\mathcal{B} \in \mathrm{CVS}(\alpha n, k)$ with total cost $\mathrm{TC}(\mathcal{B}) = \alpha \cdot \mathrm{TC}(\mathcal{A})$. In particular, since the Gray code yields a $(1, k)$-Complete Vector System with reading cost $2^k - 1$, we obtain an (n, k)-Complete Vector System with reading cost $n(2^k - 1)$.*

So far, we have established the connection between CVS and XOR-SHC and presented a general construction of CVS. However, the resulting read cost remains high. The following theorem introduces a construction technique that reduces the read cost, thereby yielding an improved upper bound.

Theorem IV.2. *Let $\mathcal{A} \in \mathrm{CVS}(n, k_1)$ and $\mathcal{B} \in \mathrm{CVS}(n, k_2)$. Then, there exists $C \in \mathrm{CVS}(n, k_1 + k_2)$ where $\mathrm{TC}(C) = 2^{k_2} \cdot \mathrm{TC}(\mathcal{A}) + \mathrm{TC}(\mathcal{B})$.*

Proof. Denote $\mathcal{A} = (A_1, \ldots, A_n)$ and $\mathcal{B} = (B_1, \ldots, B_n)$. We define $C = (C_1, \ldots, C_n)$ as follows. For $\ell \in [2^{k_1+k_2}]$, write it uniquely as $\ell = (q-1) \cdot 2^{k_1} + r$ where $q \in [2^{k_2}], r \in [2^{k_1}]$. Then, for all $j \in [n]$

$$(C_j)_\ell \triangleq \left((A_j)_r + (q-1) \cdot (A_j)_{2^{k_1}}\right) \circ \left(B_j\right)_q ,$$

where $(q-1) \cdot (A_j)_{2^{k_1}}$ means adding $(A_j)_{2^{k_1}}$ to itself $q-1$ times. We now define the map $F_C : [2^{k_1+k_2}]^n \to \mathbb{F}_2^{(k_1+k_2)n}$ by

$$F_C(x_1, \ldots, x_n) = \sum_{j=1}^{n} (C_j)_{x_j} .$$

Since the domain and codomain have the same finite cardinality, it suffices to prove that F_C is onto to show that F_C is bijective. Let $\boldsymbol{v} \in \mathbb{F}_2^{(k_1+k_2)n}$ and denote $\boldsymbol{v} = \boldsymbol{v}' \circ \boldsymbol{v}''$ where $\boldsymbol{v}' \in \mathbb{F}_2^{k_1 n}$ and $\boldsymbol{v}'' \in \mathbb{F}_2^{k_2 n}$.

Now, let $(q_1, \ldots, q_n) \in [2^{k_2}]^n$ be such that $F_{\mathcal{B}}(q_1, \ldots, q_n) = \boldsymbol{v}''$ and define

$$\boldsymbol{u}_{q_1,\ldots,q_n} = \sum_{j \in [n], q_j \text{ is odd}} (A_j)_{2^{k_1}} .$$

Let $(r_1, \ldots, r_n) \in [2^{k_1}]^n$ be such that $F_{\mathcal{A}}(r_1, \ldots, r_n) = \boldsymbol{v}' + \boldsymbol{u}$. Finally, for all $\ell \in [n]$, let $x_j = (q_j - 1) \cdot 2^{k_1} + r_j$. Then, $F_C(\boldsymbol{x}) = \sum_{j=1}^n (C_j)_{x_j} = \sum_{j=1}^n (C_j)_{(q_j-1)\cdot 2^{k_1}+r_j} = \sum_{j=1}^n \left((A_j)_{r_j} + (q_j - 1) \cdot (A_j)_{2^{k_1}}\right) \circ \left(B_j\right)_{q_j} = \left(F_{\mathcal{A}}(\boldsymbol{r}) + \sum_{j=1}^n (q_j - 1) \cdot (A_j)_{2^{k_1}}\right) \circ F_{\mathcal{B}}(\boldsymbol{q})$. Now, observe that $\sum_{j=1}^n (q_j - 1) \cdot (A_j)_{2^{k_1}} = \sum_{j \in [n], q_j \text{ is odd}} (A_j)_{2^{k_1}} = \boldsymbol{u}_{q_1,\ldots,q_n}$. Indeed, if q_j is odd then $(q_j - 1) \cdot (A_j)_{2^{k_1}} = \boldsymbol{0}$ and otherwise, $(q_j - 1) \cdot (A_j)_{2^{k_1}} = (A_j)_{2^{k_1}}$. Thus, $F_C(\boldsymbol{x}) = (\boldsymbol{v}' + \boldsymbol{u}_{q_1,\ldots,q_n}) \circ \boldsymbol{v}'' = \boldsymbol{v}$. We conclude that F_C is bijective. We are left to show the claim about the total cost. First assume that $i \in [nk_1]$. In this case, for every $j \in [n]$, and for every $\ell \in [2^{k_1+k_2}]$ where we write $\ell = (q-1)2^{k_1} + r$ such that $q \in [2^{k_2}]$ and $r \in [2^{k_1}]$ are unique, we have

$$((C_j)_\ell)_i = \begin{cases} (A_r)_i & q \text{ is odd} \\ (A_r + A_{2^{k_1}})_i & q \text{ is even} \end{cases} .$$

Thus, $\mathrm{cost}_C(i, j) = \sum_{\ell=2}^{2^{k_1+k_2}} ((C_j)_\ell)_i - ((C_j)_{\ell-1})_i = 2^{k_2} \cdot \sum_{\ell=2}^{2^{k_1}} ((A_j)_\ell)_i - ((A_j)_{\ell-1})_i = 2^{k_2} \cdot \mathrm{cost}_{\mathcal{A}}(i, j)$.

Now, assume that $i > nk_1$, and define $i' = i - nk_1 \in [nk_2]$. In this case, for every $j \in [n]$ and every $\ell \in [2^{k_1+k_2}]$, letting $q = \lceil \ell / 2^{k_1} \rceil$, we have $((C_j)_\ell)_i = ((B_j)_q)_{i'}$.

Thus, as ℓ ranges from 1 to $2^{k_1+k_2}$, the value of $((C_j)_\ell)_i$ remains constant within each block of size 2^{k_1}, and may change only when q increments by one, that is, when ℓ is a multiple of 2^{k_1}. Moreover, such a change occurs if and only if $((B_j)_{q-1})_{i'} \neq ((B_j)_q)_{i'}$. Therefore, in this case, $\mathrm{cost}_C(i, j) = \mathrm{cost}_{\mathcal{B}}(i', j)$, and the theorem follows. □

Example IV.3.

$$A_1 = (0000,\ 0010,\ 0110,\ 0111) \qquad B_1 = (00,\ 10)$$
$$A_2 = (0000,\ 1000,\ 1011,\ 0011) \qquad B_2 = (00,\ 01).$$

The sequences C_1 and C_2 constructed by the rule above are,

$$C_1 = (000000,\ 001000,\ 011000,\ 011100,\ 001110,\ 000110,\ 010110,\ 010010),$$
$$C_2 = (000000,\ 100000,\ 101100,\ 001100,\ 000001,\ 100001,\ 101101,\ 001101).$$

We now prove Theorem II.1.

Proof of Theorem II.1. According to Example IV.1, there exists $\mathcal{A} \in \mathrm{CVS}(3, 2)$ with $\mathrm{TC}(\mathcal{A}) = 10$ and by Corollary IV.1, there exists $\mathcal{B} \in \mathrm{CVS}(2, 1)$ with $\mathrm{TC}(\mathcal{B}) = 2$. Applying Theorem IV.2 with $\mathcal{A}$ and $\mathcal{B}$, we get $C \in \mathrm{CVS}(2, 4)$ with $\mathrm{TC}(C) = 2^1 \cdot 10 + 2 = 22$. Generalizing, we can apply recursively Theorem IV.2 and get the recurrence relation $a_{k+1} = 2 \cdot a_k + 2$ with $a_3 = 10$ which implies that $a_k = \frac{3}{2}2^k - 2$. Now, applying Theorem IV.1, we get an XOR-SHC$(2, k)$ with reading cost of $(\frac{3}{2}2^k - 2)/(2k)$ which implies that for all $k \geq 3$, $\mathrm{RC}^*(2, k) \leq \frac{\frac{3}{4}2^k - 1}{k}$. Now, we shall use Lemma III.2 to $\mathrm{RC}^*(n, k)$. If n is even, then we simply obtain $\mathrm{RC}^*(n, k) \leq \frac{\frac{3}{4}2^k - 1}{2}$ (we apply Lemma III.2 $n/2$ times with XOR-SHC$(2, k)$). If $n = 2m + 1$ is odd, then by applying Lemma III.2 m times with XOR-SHC$(2, k)$ and one time with XOR-SHC$(1, k)$, we get $\mathrm{RC}^*(n, k) \leq \frac{1}{2m+1}\left(2m \cdot \frac{\frac{3}{4}2^k - 1}{k} + \frac{2^k - 1}{k}\right)$ and the theorem immediately follows. □

Claim IV.1. $\mathrm{RC}^*(2, 5) \leq 4.5$ *and* $\mathrm{RC}^*(2, 6) \leq 7.5$.

Proof. Let $\mathcal{A} \in \mathrm{CVS}(2, 3)$ given in Example IV.1 and $\mathcal{B} \in \mathrm{CVS}(2, 2)$ given in Example IV.3. Applying Theorem IV.2 with $\mathcal{A}$ and $\mathcal{B}$ gives a $C \in \mathrm{CVS}(2, 5)$ with $\mathrm{TC}(C) = 2^2 \cdot \mathrm{TC}(\mathcal{A}) + \mathrm{TC}(\mathcal{B}) = 4 \cdot 10 + 5 = 45$. Consequently, by Theorem IV.1, we get an XOR-SHC$(2, 5)$ with average reading cost of 4.5.

Now, by applying Theorem IV.2 with $\mathcal{A}$ and $\mathcal{A}$ (i.e., we set $\mathcal{B} = \mathcal{A}$), we get $C \in \mathrm{CVS}(2, 6)$ with $\mathrm{TC}(C) = 90$. By Theorem IV.1, we get an XOR-SHC$(2, 6)$ with average reading cost of $90/12 = 7.5$. □

Acknowledgments

The research was funded by the European Union (ERC, 101045114). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. This work was also supported in part by the Israel Science Foundation (ISF) Grant 2462/24.

The authors thank Dr. Hironori Uchikawa for helpful discussions and for answering several questions related to this work.

References

[1] N. Shibata, H. Uchikawa, T. Shibuya, K. Nakai, K. Yanagidaira, and H. Inoue, "7-bit/2cell (x3.5), 9-bit/2cell (x4.5) nand flash memory: Half bit technology," in *2023 IEEE International Memory Workshop (IMW)*, 2023, pp. 1–4.

[2] R. L. Rivest and A. Shamir, "How to reuse a "write-once memory," *Information and control*, vol. 55, no. 1-3, pp. 1–19, 1982.

[3] J. K. Wolf, A. D. Wyner, J. Ziv, and J. Körner, "Coding for a write-once memory," *ATT Bell Laboratories Technical Journal*, vol. 63, no. 6, pp. 1089–1112, 1984.

[4] S. Kayser, E. Yaakobi, P. H. Siegel, A. Vardy, and J. K. Wolf, "Multiple-write wom-codes," in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2010, pp. 1062–1068.

[5] A. Bhatia, M. Qin, A. R. Iyengar, B. M. Kurkoski, and P. H. Siegel, "Lattice-based wom codes for multilevel flash memories," *IEEE journal on selected areas in communications*, vol. 32, no. 5, pp. 933–945, 2014.

[6] D. Burshtein and A. Strugatski, "Polar write once memory codes," *IEEE Transactions on Information Theory*, vol. 59, no. 8, pp. 5088–5101, 2013.

[7] G. Cohen, P. Godlewski, and F. Merkx, "Linear binary code for write-once memories (corresp.)," *IEEE Transactions on Information Theory*, vol. 32, no. 5, pp. 697–700, 1986.

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

[9] M. Horovitz and E. Yaakobi, "Wom codes with uninformed encoder," in *2015 IEEE Information Theory Workshop (ITW)*. IEEE, 2015, pp. 1–5.

[10] F. Merkx, "Womcodes constructed with projective geometries," *TS Traitement du signal*, vol. 1, pp. 227–231, 1984.

[11] A. Shpilka, "Capacity-achieving multiwrite wom codes," *IEEE Transactions on Information Theory*, vol. 60, no. 3, pp. 1481–1487, 2013.

[12] ——, "New constructions of wom codes using the wozencraft ensemble," *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4520–4529, 2013.

[13] Y. Wu, "Low complexity codes for writing a write-once memory twice," in *2010 IEEE International Symposium on Information Theory*. IEEE, 2010, pp. 1928–1932.

[14] Y. Wu and A. Jiang, "Position modulation code for rewriting write-once memories," *IEEE Transactions on Information Theory*, vol. 57, no. 6, pp. 3692–3697, 2011.

[15] E. Yaakobi, S. Kayser, P. H. Siegel, A. Vardy, and J. K. Wolf, "Codes for write-once memories," *IEEE Transactions on Information Theory*, vol. 58, no. 9, pp. 5985–5999, 2012.

[16] E. Sharon and I. Alrod, "Coding scheme for optimizing random i/o performance," 2012. [Online]. Available: https://arxiv.org/abs/1202.6481

[17] E. Yaakobi and R. Motwani, "Construction of random input-output codes with moderate block lengths," *IEEE Transactions on Communications*, vol. 64, no. 5, pp. 1819–1828, 2016.

[18] H. Uchikawa, N. Shibata, and T. Shibuya, "Fast readable multi-cell coding for flash memory," in *2024 IEEE International Symposium on Information Theory (ISIT)*, 2024, pp. 1149–1154.

[19] N. Shibata, H. Uchikawa, T. Shibuya, and H. Inoue, "Novel multi-level coding and architecture enabling fast random access for flash memory," in *2024 IEEE International Memory Workshop (IMW)*, 2024, pp. 1–4.

[20] J. P. Robinson and M. Cohn, "Counting sequences," *IEEE Transactions on Computers*, vol. C-30, no. 1, pp. 17–23, 1981.

[21] D. G. Wagner, "Construction of uniform gray codes," *Congressus Numerantium*, vol. 80, pp. 217–223, 1991.

[22] G. S. Bhat and C. D. Savage, "Balanced gray codes," *the electronic journal of combinatorics*, pp. R25–R25, 1996.