Codes Correcting Erasures and Deletions for Rank Modulation

Ryan Gabrys[*], **Eitan Yaakobi**[‡], **Farzad Farnoud**[†], and **Jehoshua Bruck**[†]
[*]Electrical Engineering Department, University of California, Los Angeles, Los Angeles, CA 90095, USA
[†]Electrical Engineering Department, California Institute of Technology, Pasadena, CA 91125, U.S.A
[‡]Computer Science Department, Technion - Israel Institute of Technology, Haifa 32000, Israel
rgabrys@ee.ucla.edu, {farnoud, bruck}@caltech.edu, yaakobi@cs.technion.ac.il

***Abstract*—Error-correcting codes for permutations have received a considerable attention in the past few years, especially in applications of the *rank modulation scheme* for flash memories. While several metrics have been studied like the Kendall's τ, Ulam, and Hamming distances, no recent research has been carried for erasures and deletions over permutations.**

The problems studied in this paper are motivated by a hardware implementation of the rank modulation codes. If the flash memory cells represent a permutation, which is modulated by their relative charge levels, then we explore the problems arise when some of the cells are either erased or deleted. In each case we study how these erasures and deletions affect the information carried by the remaining cells. In particular, the cells can either be *stable* and do not change their values in the permutation or *unstable* where the remaining cells form an induced permutation with less symbols. Yet another erasure model, called here *soft erasures*, assumes that all cells can be read, however the relative levels between some of the cells is not known. Our main approach in tackling these problems is to build upon the existing works of error-correcting codes in the three metrics mentioned above and leverage them in order to construct codes in each model of deletions and erasures. Lastly, we follow up on codes in the Ulam distance and improve upon the state of the art results.

I. Introduction

Flash memory has become the storage medium of choice in portable consumer electronic applications, and high performance solid-state drives (SSDs) are also being introduced into mobile computing, enterprise storage, data warehousing, and data-intensive computing systems. The rapid increase in the capacity of flash memories makes them attractive in these applications. However, the capacity increase of these technologies presents major challenges in the areas of device reliability and endurance. These challenges can be overcome through innovative coding and data handling techniques.

Flash memories are comprised of block of cells, which can store binary values or can have multiple levels and thus store more than a bit in a cell. For example, a typical block of cells in flash memories contains about 10^6 cells and the number of levels per cell can vary between 2 to 16. One of the main challenges in flash memories is to exactly program each cell to its level. In order to overcome this difficulty, ***rank modulation codes*** were proposed and studied in [6]. In this setup, the information is carried by the relative values between the cells rather than by their absolute levels. Thus, every group of cells induces a permutation, which is derived by the ranking of the level of each cell in the group. Shortly after the work in [6], several works explored codes which correct errors in permutations specifically for the rank modulation scheme; see e.g. [1], [7], [12]. These works include different metrics such as Kendall's τ, Ulam, and Hamming distances. However, none of the recent works explored the setup underneath the cells in the permutations are either deleted or erased. The goal of this work is to establish the foundations and present results for these faulty mechanisms.

The paradigms explored in this work are derived from a hardware implementation of the modulation process in rank modulation codes. In particular, while reading and comparing between the levels of the cells, it may happen that some cells are corrupted and thus cannot be read correctly. This leads to erasures in case their locations are not known or deletions otherwise. Furthermore, the missing information about the corrupted cells may or may not affect the values of the other cells. Assume that a permutation π is stored in the flash memory cells. While reading π some of the cells could be erased and/or deleted. However, while in codes over symbols, every symbol can be independently erased or deleted, for permutations, such erasures and deletions can affect the other symbols as well. To simplify our discussion, we will assume for now that only one cell was either erased or deleted. We will consider four different models, which correspond to 1) ***erasure/deletion***: whether the location of the lost cell is known, i.e. an erasure or deletion, and 2) ***stability***: whether the other symbols do or do not change their values as a result of the erasure/deletion. For example, assume that the stored permutation is $(5,3,2,4,1)$ and the third symbol 2 was either erased or deleted. For the case of stable erasure, the read information is $(5,3,?,4,1)$; for unstable erasure, the read information is $(4,2,?,3,1)$; for stable deletion, the read information is $(5,3,4,1)$; and lastly, for unstable deletion, the read information is $(4,2,3,1)$.

In this work, we discuss the first three models. The last one, which can also be extended for the case of insertions, is studied in our companion paper [4]. Our main contribution in each model of erasures and deletions is to find a known distance metric for permutations that will provide codes in the corresponding model. In particular, we show that codes based upon the Hamming distance can be used to correct stable erasures. We also show that the models of unstable erasures and stable deletions are equivalent and codes in the Ulam distance can be used in these setups.

Another model which we explore in the paper assumes that none of the cells was erased or deleted. However, it is not possible to correctly sense and thus determine the order between several cells that have adjacent levels. For example, assume that the stored permutation is $(5,3,2,4,1)$. Then, an example for a read information is $(5,\underline{32},\underline{41})$, where the notations $\underline{32}$, $\underline{41}$ indicate that the orders between 3 and 2 and between 4 and 1 are not known. However, it is known that 1 and 4 have higher ranks than 2 and 3. In this setup codes in the Kendall's τ distance are

978-1-4799-5186-4/14/$31.00 ©2014 IEEE

used in order to correct this type of erasures.

To the best of our knowledge, the research on codes combatting the proposed models is very limited. We could only specify the work by Levenshtein [9] which falls under the stable deletions model and the follow up works in [10], [11].

The rest of the paper is organized as follows. In Section II, we formally define the erasures and deletions models studied in the paper and review the Kendall's τ, Ulam, and Hamming distance metrics over permutations. In Section III, we show how to use codes in these three distance metrics in order to construct codes for our erasures and deletions models. In Section IV, we give a construction of codes in the Ulam distance which improves upon the best known ones. Due to the lack of space, some of the proofs in the paper are omitted.

II. Definitions and Preliminaries

Let $\mathbb{S}_n$ denote the set of all $n!$ permutations of n elements, chosen to be $\{1, 2, \ldots, n\}$. We use the vector notation to denote a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$. Given some permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n) \in \mathbb{S}_n$, its inverse permutation is $\pi^{-1} = (\pi_1^{-1}, \pi_2^{-1}, \ldots, \pi_n^{-1})$, where π_i^{-1} is the location of the element i in π. For example, for $\pi = (6, 1, 3, 2, 5, 4)$ we have $\pi^{-1} = (2, 4, 3, 6, 5, 1)$. The set $\{1, \ldots, n\}$ is denoted by $[n]$, and for two positive integers $a < b$, the set $\{a, \ldots, b\}$ is denoted by $[a, b]$.

We first formally define the four models of stable/unstable erasures and deletions. For a permutation $\pi = (\pi_1, \ldots, \pi_n) \in \mathbb{S}_n$, and a set of positions $I \subseteq [n]$, $\pi(I)$ is the set $\pi(I) = \{\pi_i : i \in I\}$. For an integer $a \in [n]$ and a subset $I \subseteq [n]$, the integer $a(I) \in [n]$ is defined as $a(I) = a - |\{i \in I : i < a\}|$. For example, assume $\pi = (6, 1, 3, 2, 5, 4)$ and $I = \{1, 4, 5\}$, then $\pi(I) = \{6, 2, 5\}$ and $\pi_3(\pi(I)) = 3(\{6, 2, 5\}) = 3 - 1 = 2$.

Definition 1. *Assume that $\pi = (\pi_1, \ldots, \pi_n)$ is a permutation in $\mathbb{S}_n$ and $I \subseteq \{1, \ldots, n\}$ is a positions set of size t. We consider the following four models of erasures and deletions:*

1) ***Stable Erasure** (**SE**): The permutation π suffered t stable erasures (SEs) in the positions set I, resulting in the vector $\pi' = (\pi'_1, \ldots, \pi'_n)$, if*
   a) *for all $i \in I$, $\pi'_i =$?, and*
   b) *for all $i \in [n] \setminus I$, $\pi'_i = \pi_i$.*
2) ***Unstable Erasure** (**UE**): The permutation π suffered t unstable erasures (UEs) in the positions set I, resulting in the vector $\pi' = (\pi'_1, \ldots, \pi'_n)$, if*
   a) *for all $i \in I$, $\pi'_i =$?, and*
   b) *for all $i \in [n] \setminus I$, $\pi'_i = \pi_i(\pi(I))$.*
3) ***Stable Deletion** (**SD**): The permutation π suffered t stable deletions (SDs) in the positions set I, resulting in the vector $\pi' = (\pi'_1, \ldots, \pi'_{n-t})$, if*
   for all $k \in [n] \setminus I$ and $i = k(I)$, $\pi'_i = \pi_k$.
4) ***Unstable Deletion** (**UD**): The permutation π suffered t unstable deletions (UDs) in the positions set I, resulting in the permutation $\pi' = (\pi'_1, \ldots, \pi'_{n-t}) \in \mathbb{S}_{n-t}$, if*
   for all $k \in [n] \setminus I$ and $i = k(I)$, $\pi'_i = \pi_k(\pi(I))$.

*A code $\mathcal{C} \subseteq \mathbb{S}_n$ is called a **t-SE/UE/SD/UD-correcting code** if it can correct at most t SEs/UEs/SDs/UDs, respectively.*

The next example illustrates these four models.

Example 1. *Let $\pi = (6, 1, 3, 2, 5, 4) \in \mathbb{S}_6$ and $I = \{1, 4, 5\}$. Then, the following vectors are the received ones for each model:*

1) *Stable Erasure: $\pi' = (?, 1, 3, ?, ?, 4)$.*
2) *Unstable Erasure: $\pi' = (?, 1, 2, ?, ?, 3)$.*
3) *Stable Deletion: $\pi' = (1, 3, 4)$.*
4) *Unstable Deletion: $\pi' = (1, 2, 3)$.*

Note that the first model in Definition 1 is the easiest one and the last one is the hardest one, with respect to the amount of information that is lost. The last model of unstable deletions can be generalized for insertions such that more symbols can be inserted and the other cells scale their value accordingly. For example, assume that the symbol 2 is inserted between the first and second cells in the permutation π from Example 1. Then, the read permutation in $\mathbb{S}_7$ will be $(7, 2, 1, 4, 3, 6, 5)$. This model of insertions as well as the fourth model of UDs are addressed and studied in our work in [4].

The reading errors in Definition 1 assume that a certain symbol was not read properly and thus was either deleted or erased. The next studied model assumes that all symbols are read and received, however some symbols are read together so the order between them, and only them, is not known. We define this erasure model formally.

Definition 2. *We say that a permutation $\pi \in \mathbb{S}_n$ suffers an $\boldsymbol{e} = (e_1, \ldots, e_t)$-**soft-erasure** if there are t sets $I_1, \ldots, I_t$, such that for $1 \leqslant \ell \leqslant t$, $I_\ell = [a_\ell, b_\ell]$, $b_\ell = a_\ell + e_\ell - 1$, $1 \leqslant a_1 < b_1 < a_2 < b_2 < \cdots < a_t < b_t \leqslant n$, and only the orders of the elements in each of the t groups $\pi(I_1), \ldots, \pi(I_t)$ are not known. The **erasure-weight** of the soft-erasure $\boldsymbol{e}$ is $\omega(\boldsymbol{e}) = \sum_{\ell=1}^{t} \binom{e_\ell}{2}$. A code $\mathcal{C} \subseteq \mathbb{S}_n$ is called an E-**soft-erasure-correcting code** if it can correct any $\boldsymbol{e}$-soft-erasure of erasure-weight at most E.*

The motivation in choosing the erasure-weight terminology results from the observation that this number indicates the number of element pairs which their order is not known. Our approach in finding codes for the aforementioned models is to use, when possible, some of the already existing results of error-correcting codes over permutations. There are several different metrics that these codes were studied for and the following metrics are the ones we use in this work.

An adjacent transposition in a permutation $\pi \in \mathbb{S}_n$ is the local exchange of two adjacent elements in π. The ***Kendall's τ*** distance [8] between two permutations $\sigma, \pi \in \mathbb{S}_n$ is denoted by $d_\tau(\sigma, \pi)$ and is defined to be the minimum number of adjacent transpositions required to obtain the permutation π from the permutation σ. The ***Hamming distance*** between two permutations $\pi, \sigma \in \mathbb{S}_n$, denoted by $d_H(\pi, \sigma)$, is defined as the number of positions for which π and σ differ. For two permutations $\pi, \sigma \in \mathbb{S}_n$, let $\ell(\pi, \sigma)$ be the length of a longest common subsequence of π and σ. The ***Ulam distance*** between π and σ is defined as $d_\circ(\pi, \sigma) = n - \ell(\pi, \sigma)$ [2], [3], [5].

Example 2. *Let $\pi = (4, 3, 1, 2, 5)$ and $\sigma = (4, 3, 5, 1, 2)$, then $d_\tau(\pi, \sigma) = 2$, $d_H(\pi, \sigma) = 3$, and $d_\circ(\pi, \sigma) = 1$.*

The minimum distance of a code, according to any metric, is the minimum distance between every two codewords in the code. For a code $\mathcal{C} \subseteq \mathbb{S}_n$, its minimum Kendall's τ, Hamming, Ulam distance is denoted by $d_\tau(\mathcal{C})$, $d_H(\mathcal{C})$, $d_\circ(\mathcal{C})$, respectively.

III. Codes from Existing Codes in other Metrics

In this section we present codes for the erasures and deletions models defined in Section II. In particular, we show how codes in the Kendall's τ, Hamming, and Ulam distance can be used for these models.

Let us start with stable erasures. First notice that if only a single stable erasure occurred then it is immediate to complete the missing symbol since its location is known by the erasure as well as its value, which is the missing symbol in the permutation. Thus, the code $\mathbb{S}_n$ is a single-SE-correcting code. For more than a single stable erasure, we show in the next theorem how codes in the Hamming distance are necessary and sufficient in the SE model.

Theorem 3. *A code $\mathcal{C} \subseteq \mathbb{S}_n$ is a t-SE-correcting code if and only if $d_H(\mathcal{C}) \geqslant t+1$.*

Proof: We show that if $d_H(\mathcal{C}) \geqslant t+1$ then $\mathcal{C}$ is a t-SE-correcting code. Assume in the contrary that $\mathcal{C}$ is not a t-SE-correcting code. Thus, there are two permutations $\pi, \sigma \in \mathcal{C}$ and two positions sets $I_1, I_2 \subseteq [n]$, each of size at most t, such that if π' is the result of stable erasures in the positions set I_1 in π and σ' is the result of stable erasures in the positions set I_2 in σ, then $\pi' = \sigma'$. First notice that $I_1 = I_2$. Otherwise, assume without loss of generality that there exists $i \in I_1 \setminus I_2$, so we get $\pi'_i =?$ and $\sigma'_i \neq ?$ which is a contradiction. Hence, we can denote $I_1 = I_2 = I$ and $|I| \leqslant t$. Then we get that for every $i \in [n] \setminus I$, $\pi_i = \pi'_i = \sigma'_i = \sigma_i$, and thus $d_H(\pi, \sigma) \leqslant |I| \leqslant t$, in contradiction.

The second part holds since we can consider $\mathcal{C}$ as a code in $[n]^n$ so its minimum Hamming distance has to be $t+1$. ∎

We next move to the models of unstable erasures and stable deletions. Note that in unstable erasures the locations but not the values are not known, while in stable deletions the values but not locations are not known. The next lemma establishes a property claiming that these two models are equivalent.

Lemma 4. *A permutation $\pi \in \mathbb{S}_n$ suffered t UEs if and only if its inverse permutation π^{-1} suffered t SDs.*

Proof: Let $\pi = (\pi_1, \ldots, \pi_n)$ and $\pi^{-1} = (\pi_1^{-1}, \ldots, \pi_n^{-1})$. Assume that π suffered t UEs in the positions set I, resulting in the vector $\pi' = (\pi'_1, \ldots, \pi'_n)$. Let $\pi'^{-1} = (\pi'^{-1}_1, \ldots, \pi'^{-1}_{n-t})$ be a length-$(n-t)$ vector which specifies the locations of the $(n-t)$ non-erased symbols in π'. For every $k \in [n] \setminus I$, $\pi'_k = \pi_k(\pi(I))$ and hence the location of $i \in [n-t]$ in π' is the location of the symbol k_i in π such that $i = k_i(\pi(I))$. That is, for $i \in [n-t]$, $\pi'^{-1}_i = \pi^{-1}_{k_i}$, where $i = k_i(\pi(I))$. Therefore, π'^{-1} is the vector received from π^{-1} after having t stable deletions in the positions set $\pi(I)$.

To prove the other direction, since $(\pi^{-1})^{-1} = \pi$, we can simply prove that if π suffered t SDs then π^{-1} suffered t UEs. If so, assume that π suffered t SDs in the positions set I, resulting with the vector $\pi' = (\pi'_1, \ldots, \pi'_{n-t})$, where for $i \in [n-t]$, $\pi'_i = \pi_{k_i}$, where $i = k_i(I)$. Let $\pi'^{-1} = (\pi'^{-1}_1, \ldots, \pi'^{-1}_n)$ be a length-n vector which specifies the locations of the symbols $[n]$ in π' or specifies a ? in case that the symbol does not appear in π'. That is, for $i \in \pi(I)$ have $\pi'^{-1}_i =?$ and for $i \notin \pi(I)$, $\pi'^{-1}_i = \pi_i^{-1}(I)$, or, since $\pi^{-1}(\pi(I)) = I$, $\pi'^{-1}_i = \pi_i^{-1}(\pi^{-1}(\pi(I)))$. Hence, the vector π'^{-1} equals the vector received from π^{-1} after having t UEs in the positions set $\pi(I)$. ∎

As a result of Lemma 4 we conclude the following corollary.

Corollary 5. *There exists a t-UE-correcting code of cardinality M if and only if there exists a t-SD-correcting code of cardinality M.*

To complete this discussion we only need to consider codes in one of these two models. We will show how codes in the Ulam distance are necessary and sufficient for codes in the SD model.

Theorem 6. *A code $\mathcal{C} \subseteq \mathbb{S}_n$ is a t-SD-correcting code if and only if $d_\circ(\mathcal{C}) \geqslant t+1$.*

Proof: We show that if $\mathcal{C}$ is a t-SD-correcting code then $d_\circ(\mathcal{C}) \geqslant t+1$. Assume in the contrary that $d_\circ(\mathcal{C}) \leqslant t$ and let $\pi, \sigma \in \mathcal{C}$ be such that $d_\circ(\pi, \sigma) = t' \leqslant t$. Hence, π and σ have a common subsequence of length $\ell(\pi, \sigma) = n - d_\circ(\pi, \sigma) = n - t' \geqslant n - t$ and let S be the set of symbols in this common subsequence. Let I_π be the positions of the symbols $[n] \setminus S$ in π and similarly let I_σ be the positions of the symbols $[n] \setminus S$ in σ. Then, if π' is the result of t SDs in π in the positions set I_π and σ' is the result of t SDs in σ in the positions set I_σ, we get that $\pi' = \sigma'$. Therefore, the code $\mathcal{C}$ is not a t-SD-correcting code, which is a contradiction.

In order to prove the other direction, we show that if $d_\circ(\mathcal{C}) \geqslant t+1$ then $\mathcal{C}$ is a t-SD-correcting code. Assume in the contrary that $\mathcal{C}$ is not a t-SD-correcting code. Thus, there are two permutations $\pi, \sigma \in \mathcal{C}$ and two positions sets $I_1, I_2 \subseteq [n]$, each of size at most t, such that if π' is the result of stable deletions in the positions set I_1 in π and σ' is the result of stable deletions in the positions set I_2 in σ, then $\pi' = \sigma'$. First we have that $|I_1| = |I_2|$ and since $\pi' = \sigma'$, π and σ have a common subsequence of length $n - |I_1| \geqslant n - t$. Therefore, $d_\circ(\pi, \sigma) \leqslant n - (n-t) = t$, in contradiction again. ∎

Lastly in this section we turn to handle the soft-erasure model. The connection between this model of erasures and the Kendall's τ-metric is established in the next theorem.

Theorem 7. *Let $\mathcal{C} \subseteq \mathbb{S}_n$ be a code where $d_\tau(\mathcal{C}) \geqslant E+1$. Then, $\mathcal{C}$ is an E-soft-erasure-correcting code.*

Proof: Assume to the contrary that $\mathcal{C}$ is not an E-soft-erasure-correcting code. Then, there exist two permutations π, σ, and two soft-erasures $\boldsymbol{e}^\pi = (e_1^\pi, \ldots, e_t^\pi)$, $\boldsymbol{e}^\sigma = (e_1^\sigma, \ldots, e_t^\sigma)$ of erasure-weights at most E such that the vectors received after each of the two soft-erasures are the same. Let $I_1^\pi, \ldots, I_t^\pi$ be the sets of erasures in π and similarly, $I_1^\sigma, \ldots, I_t^\sigma$ are the sets of erasures in σ. First note that $I_j^\pi = I_j^\sigma$ for $1 \leqslant j \leqslant t$, since otherwise the received vectors, as a result of the soft-erasures, are not the same. From the same reason, the symbols in each of the t groups are the same for π and σ and the order of all other symbols in π and σ is the same. Therefore, there can be only $\sum_{\ell=1}^t \binom{e_\ell^\pi}{2}$ pairs of symbols that π and σ do not agree on. However, that means that the Kendall's τ distance between π and σ is at most $\sum_{\ell=1}^t \binom{e_\ell^\pi}{2} \leqslant E$, which is a contradiction. ∎

IV. New Codes for the Ulam Metric

In this section, we present new codes for the Ulam metric. In the first subsection, we introduce some notation, tools, and codes that will be used in the following subsection to describe the code construction for the Ulam metric. We proceed by describing a code over $\mathbb{S}_n$ that can correct a prescribed number of stable deletions. As a consequence of Theorem 6, these codes are also suitable for the Ulam metric. Lastly, we will comment on how our construction improves upon the state of the art codes for this metric from [5]. For the remainder of this section, we use the terms deletion(s) and stable deletion(s) interchangeably.

Furthermore, we use the terms erasure(s) and stable erasure(s) interchangeably as well.

A. Notation and Auxiliary Codes

For a sequence π with elements from the set $[n] \cup \{?\}$ and for $s \in [n]$, let $D(\pi, s)$ be the result of removing all occurrences of the symbol s in π. If s is not contained in π, then $D(\pi, s) = \pi$. More generally, for $\mathcal{I} \subset [n]$, $D(\pi, \mathcal{I})$ is the result of removing all symbols from $\mathcal{I}$ in π. Notice that in this case, the set $\mathcal{I}$ contains the symbols to be deleted and not the locations of those symbols. For $\mathcal{I}' \subset [n]$, let $\sigma = Er(\pi, \mathcal{I}')$ be the result of substituting the elements of $\mathcal{I}'$ in π with the symbol ?.

The following code can determine the locations of deletions given that the locations of the deletions satisfy certain constraints, that will be explained later. For the remainder of this section, we assume that n, ℓ are positive integers where $n > \ell$ and $\ell | n$. We define

$$\mathcal{C}^L_{\ell,n} = \{\pi \in \mathbb{S}_n : i \in [n],\ \pi_i \equiv i - 1 (\mathrm{mod}\, \ell)\}.$$

For $\sigma = (\sigma_1, \ldots, \sigma_n) \in \mathbb{S}_n$, the vector $\sigma(\mathrm{mod}\ell) = (\sigma'_1, \ldots, \sigma'_n)$ is defined by $\sigma'_i = \sigma_i (\mathrm{mod}\, \ell)$ for $1 \leqslant i \leqslant n$. Thus, for any codeword $\pi \in \mathcal{C}^L_{\ell,n}$, the vector $\pi(\mathrm{mod}\ell)$ is a periodic sequence of length n and period ℓ where each period is $(0, \ldots, \ell - 1)$.

We now describe a decoding algorithm $\mathcal{D}^L_{\ell,n}$ for the code $\mathcal{C}^L_{\ell,n}$. We note that although the map $\mathcal{D}^L_{\ell,n}$ is not formally a decoder, with a slight abuse of notation it will be referred to as such and the procedure below will be referred to as a decoding algorithm. Suppose that $\pi \in \mathcal{C}^L_{\ell,n}$ is the stored codeword and σ is the retrieved word, where $\sigma = D(\pi, \mathcal{I})$ with $\mathcal{I} \subset [n]$, $|\mathcal{I}| = t < n$. The input to $\mathcal{D}^L_{\ell,n}$ is σ.

1) Initialize $j = 0$ and let $\zeta^{(1)} = (\sigma, 0)$.
2) Let $j = j + 1$.
3) Let i_j be the smallest i, where $1 \leqslant i \leqslant n - t + j$, such that $\zeta^{(j)}_i \not\equiv i - 1 (\mathrm{mod}\, \ell)$ and $\zeta^{(j)}_i \neq ?$. If no such symbol exists, go to step 5).
4) Let $\zeta^{(j+1)}$ be the result of inserting the symbol ? into $\zeta^{(j)}$ at position i_j. Go to step 2).
5) Define $\hat{\pi} = (\zeta^{(j)}_1, \ldots, \zeta^{(j)}_{n-t+j-1})$.

Example 3. *Suppose* $n = 12$ *and* $\ell = 3$ *and let* $\pi = (3,7,5,9,1,8,6,4,2,12,10,11) \in \mathcal{C}^L_{3,12}$ *and let* $\mathcal{I} = \{1,3,8,11\}$. *Then,* $\sigma = (7,5,9,6,4,2,12,10)$. *In this case,* $\zeta^{(1)} = (7,5,9,6,4,2,12,10,0)$, $\zeta^{(2)} = (?,7,5,9,6,4,2,12,10,0)$, $\zeta^{(3)} = (?,7,5,9,?,6,4,2,12,10,0)$, $\zeta^{(4)} = (?,7,5,9,?,?,6,4,2,12,10,0)$, *and* $\zeta^{(5)} = (?,7,5,9,?,?,6,4,2,12,10,?,0)$, *and* $\hat{\pi} = (?,7,5,9,?,?,6,4,2,12,10,?)$.

Let $b(\pi, \mathcal{I}, \ell)$ be equal to 1 if there exists a subset of ℓ symbols from $\mathcal{I}$ that appear consecutively in π. Otherwise, $b(\pi, \mathcal{I}, \ell) = 0$. If $b(\pi, \mathcal{I}, \ell) = 1$, then let $\ell(\pi, \mathcal{I})$ be the set of symbols that constitute the first occurrence of a subset of ℓ symbols from $\mathcal{I}$ that appear consecutively in π. If $b(\pi, \mathcal{I}, \ell) = 0$, then let $\ell(\pi, \mathcal{I}) = \emptyset$. For example, let π be as in Example 3, then, $b(\pi, \{3,7,5,8,6,4\}, 3) = 1$ and $3(\pi, \{3,7,5,8,6,4\}) = \{3,5,7\}$.

The following claim follows from the decoding algorithm for $\mathcal{C}^L_{\ell,n}$. For a sequence $\boldsymbol{x}$ with symbols from $[n] \cup \{?\}$ let $|\boldsymbol{x}|$ denote the length of $\boldsymbol{x}$.

Claim 8. *For positive integers* n, ℓ, *suppose* $\pi \in \mathcal{C}^L_{\ell,n}$, $\mathcal{I} \subset [n]$, *and* $\sigma = D(\pi, \mathcal{I})$. *If* $b(\pi, \mathcal{I}, \ell) = 0$, *then* $\mathcal{D}^L_{\ell,n}(\sigma) = Er(\pi, \mathcal{I})$. *Otherwise,* $|\mathcal{D}^L_{\ell,n}(\sigma)| < |\pi| = n$.

In words, if the longest maximal substring deleted from π has length less than ℓ, that is $b(\pi, \mathcal{I}, \ell) = 0$, then the output of the decoder is π with symbols of $\mathcal{I}$ replaced by ?, which provides the locations of the deleted symbols. Otherwise, the decoder returns a sequence with length shorter than the length of π.

For our code construction in the next subsection we will use two more codes which are described as follows. The first code is a code capable of correcting a single stable deletion over permutations. Let $\mathcal{C}^E_{n'} \subseteq \mathbb{S}_{n'}$ be a code that can correct a single stable deletion from [9] and let $\mathcal{D}^E_{n'}$ be a decoder for this code [9]. The decoder $\mathcal{D}^E_{n'}$ operates as follows. Suppose $\sigma = D(\pi, s)$, where $\pi \in \mathcal{C}^E_{n'}$. The output of $\mathcal{D}^E_{n'}(\sigma)$ is the ordered triplet (s, s', s'') where s is the symbol deleted from π to obtain σ, s' is the symbol immediately before s in π, and s'' is the symbol immediately after s in π. If s is the final symbol in π, then $s'' = 0$ and similarly if s is the first symbol in π, then $s' = 0$.

The second code will be a code in the Hamming metric. Let $\mathcal{C}^H_{d,n} \subseteq \mathbb{S}_n$ be a code with minimum Hamming distance d. From Theorem 3, there exists a decoder $\mathcal{D}^H_{d,n} : ([n] \cup \{?\})^n \to \mathbb{S}_n$ for $\mathcal{C}^H_{d,n}$ that can correct up to $d - 1$ stable erasures, where the location of the erasures are represented by the symbol ?.

B. Code Construction

Using the tools from the previous subsection, we now present a code capable of correcting $2\ell - 1$ stable deletions. The idea will be to combine the constraints for the codes $\mathcal{C}^L_{\ell,n}$, $\mathcal{C}^E_{n/\ell}$, and $\mathcal{C}^H_{3\ell-2,n}$. For a permutation $\pi \in \mathbb{S}_n$ and an integer $0 \leqslant i \leqslant \ell - 1$, let $\pi_{\pi \equiv_\ell i}$ be the subsequence of π that only contains the symbols from the set $\{s \in [n] : s \equiv i (\mathrm{mod}\, \ell)\}$. For integers m, n, k, and $\boldsymbol{x} \in [n]^m$, let $(\boldsymbol{x} - k)/\ell = (y_1, \ldots, y_m)$ be such that for $1 \leqslant i \leqslant m$, $y_i = \lfloor (x_i - k)/\ell \rfloor$.

Construction 1. *For positive integers* n, ℓ *where* $n > \ell$ *and* $\ell | n$, *let* $\mathcal{C}^U_{2\ell,n} \subseteq \mathbb{S}_n$ *be the code consisting of all permutations* $\pi \in \mathbb{S}_n$ *that satisfy the following conditions:*

1) $\pi \in \mathcal{C}^H_{3\ell-2,n}$
2) $\pi \in \mathcal{C}^L_{\ell,n}$, *and*
3) $(\pi_{\pi \equiv_\ell i} - i)/\ell \in \mathcal{C}^E_{n/\ell}$ *for* $0 \leqslant i \leqslant \ell - 1$.

We will show that the code $\mathcal{C}^U_{2\ell,n}$ has Ulam distance at least 2ℓ by showing that it can recover from any $m = 2\ell - 1$ deletions. Suppose $\sigma = D(\pi, \mathcal{I}) \in [n]^{n-m}$ where $\mathcal{I} \subset [n]$, $|\mathcal{I}| = m$, and $\pi \in \mathcal{C}^U_{2\ell,n}$.

We first outline the decoding procedure. We will first attempt to determine the locations of the deletions using the decoder $\mathcal{D}^L_{\ell,n}$. From Claim 8, $\mathcal{D}^L_{\ell,n}$ can determine the locations of all the deletions except if $b(\pi, \mathcal{I}, \ell) = 1$. Recall that if $b(\pi, \mathcal{I}, \ell) = 1$, then there was a substring deleted from π whose length is at least ℓ. In this case, the decoder $\mathcal{C}^E_{n/\ell}$ is used to determine the location where the substring (of length at least ℓ) was deleted from π and the locations of the remaining deletions are discovered with the aid of $\mathcal{D}^L_{\ell,n}$. Finally, using $\mathcal{D}^H_{3\ell-2,n}$ the values of the deleted symbols are recovered.

We now turn to formally define the decoding procedure. We refer to the decoding map for $\mathcal{C}^U_{2\ell,n}$ as $\mathcal{D}^U_{2\ell,n} : [n]^{n-m} \to \mathbb{S}_n$. The input to the map is σ and the output is an estimate $\hat{\pi}$ of the codeword $\pi \in \mathcal{C}^U_{2\ell,n}$. The decoder is presented in Alg. 1, where we use the following notation. Let $s', s'' \in [n]$. For a vector $\sigma \in [n]^{n-m}$, we refer to the set of symbols in σ that are

between the symbols s', s'', exclusive, as $\sigma[s', s'']$. If $s' = 0$, then $\sigma[s', s'']$ is the set of symbols that appear before the symbol s'' in σ. Similarly, if $s'' = 0$, then $\sigma[s', s'']$ is the set of symbols that appear after the symbol s' in σ.

Algorithm 1: $\mathcal{D}^U_{2\ell,n} : [n]^{n-m} \to \mathbb{S}_n$

input : the retrieved permutation $\sigma = D(\pi, \mathcal{I})$
output: estimate of π, $\hat{\pi}$

1 $\sigma^{(1)} \longleftarrow \mathcal{D}^L_{\ell,n}(\sigma)$;
2 **if** $|\sigma^{(1)}| = n$ **then**
3 $\quad \sigma^{(4)} \longleftarrow \sigma^{(1)}$;
4 **else**
5 $\quad k \longleftarrow \min\{x \in \mathbb{Z}_\ell : |\sigma_{\sigma \equiv_\ell x}| = \frac{n}{\ell} - 1\}$;
6 $\quad (s, s', s'') \longleftarrow \mathcal{D}^E_{n/\ell}\left((\sigma_{\sigma \equiv_\ell k} - k)/\ell\right)$;
7 $\quad \mathcal{S} = \sigma[\ell \cdot s' + k, \ell \cdot s'' + k]$;
8 $\quad \sigma^{(2)} \longleftarrow D(\sigma, \mathcal{S})$;
9 $\quad \sigma^{(3)} \longleftarrow$ result of inserting $\ell \cdot s + k$ between $\ell \cdot s' + k$ and $\ell \cdot s'' + k$ in $\sigma^{(2)}$;
10 $\quad \sigma^{(4)} \longleftarrow \mathcal{D}^L_{\ell,n}(\sigma^{(3)})$;
11 **end**
12 $\hat{\pi} \longleftarrow \mathcal{D}^H_{3\ell-2,n}(\sigma^{(4)})$;

Theorem 9. *The code $\mathcal{C}^U_{2\ell,n}$ has Ulam distance at least 2ℓ.*

Proof: The result is proven by showing that the output $\hat{\pi}$ of Alg. 1 equals π. In this proof, let $A = \ell(\pi, \mathcal{I})$. Suppose that at step 2, we have $|\sigma^{(1)}| = n$. Then, from Claim 8, we have $\sigma^{(1)} = Er(\pi, \mathcal{I})$ and $A = \emptyset$. In step 12, since π belongs to a code with minimum Hamming distance $3\ell - 2$, and $|\mathcal{I}| = 2\ell - 1 \leqslant 3\ell - 2$, the values of the deleted symbols can be recovered. Thus, when $|\sigma^{(1)}| = n$, we have that $\hat{\pi} = \pi$.

We now suppose that $|\sigma^{(1)}| = n - \ell$, which is the only other possibility for if $A \neq \emptyset$, then $|A| = \ell$. Since $2\ell - 1$ elements are deleted from π, by the pigeon hole principle, there exists k such that $|\sigma_{\sigma \equiv_\ell k}| = \frac{n}{\ell} - 1$ (the algorithm arbitrarily picks the smallest possible value for k in step 5). The deleted symbol from $\pi_{\pi \equiv_\ell k}$, that is the only deleted symbol that is equal to $k (\mathrm{mod}\, \ell)$, is $\ell \cdot s + k$, where $(s, s', s'') = \mathcal{D}^E_{n/\ell}(\frac{\sigma_{\sigma \equiv_\ell k} - k}{\ell})$. For simplicity of presentation, in the algorithm and in this discussion we ignore the possibility that $s' = 0$ or $s'' = 0$; these cases can be handled similarly.

Since the ℓ symbols of A are consecutive in π, there is one element of A that is equal to $k (\mathrm{mod}\, \ell)$. But since $\ell \cdot s + k$ is the only deleted element from π that is equal to k modulo ℓ, we have $\ell \cdot s + k \in A$. This in turn implies that the symbols of A are located between the symbol $\ell \cdot s' + k$ and the symbol $\ell \cdot s'' + k$ in π. So the size of the set $\mathcal{S}$ in step 8 is at most $(2\ell - 1) - \ell = \ell - 1$, where $2\ell - 1$ is the number of symbols between $\ell \cdot s' + k$ and $\ell \cdot s'' + k$ in π and ℓ is the number of symbols in A. Hence, $|\sigma^{(2)}| = n - |\mathcal{I}| - |\mathcal{S}| \leqslant n - (3\ell - 2)$.

In step 9 of the algorithm, $\ell \cdot s + k$ is inserted in its correct position. So now there are only $3\ell - 3$ elements missing from $\sigma^{(3)}$ compared to π. In other words, there is a set $\mathcal{I}'$ such that $\sigma^{(3)} = D(\pi, \mathcal{I}')$, where $|\mathcal{I}'| \leqslant 3\ell - 3$. Since $\ell \cdot s + k \notin \mathcal{I}'$, we have $b(\pi, \mathcal{I}', \ell) = 0$. Hence, by Claim 8, $\sigma^{(4)} = Er(\pi, \mathcal{I}')$ at step 10. The decoder $\mathcal{D}^H_{3\ell-2,n}$ can recover π from $\sigma^{(4)}$ in step 12 since $|\mathcal{I}'| \leqslant 3\ell - 3$ and the minimum Hamming distance of the code $C^U_{2\ell,n}$ is $3\ell - 2$. ∎

We illustrate Algorithm 1 with the following example.

Example 4. *Suppose $\pi = (9, 7, \mathbf{8}, \mathbf{6}, \mathbf{1}, 11, 3, \mathbf{10}, 2, 12, \mathbf{4}, 5) \in \mathcal{C}^U_{6,12}$ so that $\ell = 3$. Suppose $\sigma = D(\pi, \mathcal{I}) = (9, 7, 11, 3, 2, 12, 5)$ where $\mathcal{I} = \{1, 4, 6, 8, 10\}$. We will show that, if Algorithm 1 is invoked with σ as input, then $\hat{\pi} = \pi$.*

At step 1, we have $\sigma^{(1)} = (9, 7, 11, 3, ?, 2, 12, ?, 5)$. Then since $|\sigma^{(1)}| = 9 < 12$, we proceed to step 5. At step 5, $k = 0$ since $|\sigma_{\sigma \equiv_3 0}| = |(9, 3, 12)| = 4 - 1$. Then at step 6, we the output of $\mathcal{D}^E_4$ would be $(s, s', s'') = (2, 3, 1)$. At step 7, the set $\mathcal{S} = \sigma[9, 3] = \{7, 11\}$. Notice that $|\mathcal{S}| = 2$ in this case. After the elements from the set $\mathcal{S}$ are removed from σ, at step 8 we have $\sigma^{(2)} = (9, 3, 2, 12, 5)$. At step 9, we insert the symbol 6 into $\sigma^{(2)}$ giving that $\sigma^{(3)} = (9, 6, 3, 2, 12, 5)$. At step 10, $\sigma^{(4)} = (9, ?, ?, 6, ?, ?, 3, ?, 2, 12, ?, 5)$. Since there are 6 total erasures and $\mathcal{D}^H_{7,12}$ is the decoder for a code with Hamming distance 7, the output of $\mathcal{D}^H_{7,12}$ at step 12 will be $\hat{\pi} = (9, 7, 8, 6, 1, 11, 3, 10, 2, 12, 4, 5) = \pi$ as desired.

We now briefly compare a code created according to Construction 1 to the codes from [5]. A code $\mathcal{C}^U_{2\ell,n}$ with Ulam distance 2ℓ requires interleaving ℓ subsequences as a consequence of item 1) in Construction 1. We note that, if the codes from [5] were adopted, then constructing a code with Ulam distance 2ℓ would require interleaving at least $2(\ell - 1) + 1$ subsequences. Because Construction 1 requires the interleaving of fewer subsequences, it can be shown that for large n Construction 1 produces codebooks with much higher cardinalities than the codes presented in [5]. A more rigorous comparison between the code constructions are left for an extended version of the paper.

Acknowledgment

The work of Eitan Yaakobi, Farzad Farnoud, and Jehoshua Bruck was supported in part by Intellectual Ventures, an NSF grant CIF-1218005, and the U.S.-Israel Binational Science Foundation, Jerusalem, Israel, under Grant No. 2010075. Ryan Gabrys was supported by the SMART scholarship.

References

[1] A. Barg and A. Mazumdar, "Codes in permutations and error correction for rank modulation," *IEEE Trans. on Information Theory*, vol. 56, no. 7, pp. 3158–3165, Jul. 2010.

[2] W.A. Beyer, M.L. Stein, and S.M. Ulam, "Metric in Biology, an Introduction," Preprint LA-4973, Univ. of Calif., Los Alamos, 1972.

[3] M. Deza and T. Huang, "Metrics on permutations, a survey," *J. Combin. Inf. Syst. Sci.*, vol. 23, pp. 173–185, 1998.

[4] R. Gabrys, E. Yaakobi, F. Farnoud, F. Sala, L. Dolecek, and J. Bruck, "Single-deletion-correcting codes over permutations," submitted to *IEEE International Symposium on Information Theory*, Honolulu, HI, Jun.-Jul. 2014.

[5] F. Farnoud (Hassanzadeh), V. Skachek, and O. Milenkovic, "Error-correction in flash memories via codes in the Ulam metric," *IEEE Trans. Information Theory*, vol. 59, no. 5, pp. 3003–3020, May 2013.

[6] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation for flash memories," *IEEE Trans. on Information Theory*, vol. 55, no. 6, pp. 2659–2673, Jun. 2009.

[7] A. Jiang, M. Schwartz, and J. Bruck, "Correcting charge-constrained errors in the rank-modulation scheme," *IEEE Trans. on Information Theory*, vol. 56, no. 5, pp. 2112–2120, May 2010.

[8] M. Kendall and J.D. Gibbons, *Rank Correlation Methods*. New York: Oxford Univ. Press, 1990.

[9] V. I. Levenshtein, "On perfect codes in deletion and insertion metric," *Discretnaya Mathematika*, vol. 3, no. 1, pp. 3–20, 1991.

[10] G.M. Tenengolts, "Nonbinary codes, correcting single deletion or insertion (Corresp.)," *IEEE Trans. Information Theory*, vol. 30, no. 5, pp. 766–769, Sep. 1984.

[11] R.R. Varshamov and G.M. Tenengolts, "Codes which correct single asymmetric errors," *Avtomatika i Telemekhanika*, vol. 6, no. 2, pp. 288–292, 1965.

[12] H. Zhou, A. Jiang, and J. Bruck, "Systematic error-correction codes for rank modulation," *Proc. IEEE International Symposium on Information Theory*, pp. 2978–2982, Cambridge, MA, Jul. 2012.