Constructions of Snake-in-the-Box Codes for Rank Modulation

Michal Horovitz and Tuvi Etzion, *Fellow, IEEE*

***Abstract*— Snake-in-the-box code is a Gray code, which is capable of detecting a single error. Gray codes are important in the context of the rank modulation scheme, which was suggested recently for representing information in flash memories. For a Gray code in this scheme, the codewords are permutations, two consecutive codewords are obtained using the push-to-the-top operation, and distance measure is defined on permutations. In this paper, the Kendall's τ-metric is used as the distance measure. We present a general method for constructing such Gray codes. We apply the method recursively to obtain a snake of length $M_{2n+1} = ((2n+1)(2n)-1)M_{2n-1}$ for permutations of S_{2n+1}, from a snake of length M_{2n-1} for permutations of S_{2n-1}. Thus, we have $\lim_{n\to\infty} M_{2n+1}/S_{2n+1} \approx 0.4338$, improving on the previous known ratio of $\lim_{n\to\infty} 1/\sqrt{(\pi n)}$. Using the general method, we also present a direct construction. This direct construction is based on necklaces and it might yield snakes of length $(2n+1)!/2-2n+1$ for permutations of S_{2n+1}. The direct construction was applied successfully for S_7 and S_9, and hence $\lim_{n\to\infty} M_{2n+1}/S_{2n+1} \approx 0.4743$.**

***Index Terms*— Flash memory, Gray code, necklaces, push-to-the-top, rank modulation scheme, snake-in-the-box code, spanning tree, 3-uniform hypergraph.**

I. Introduction

FLASH memory is a non-volatile technology that is both electrically programmable and electrically erasable. It incorporates a set of cells maintained at a set of levels of charge to encode information. While raising the charge level of a cell is an easy operation, reducing the charge level requires the erasure of the whole block to which the cell belongs. For this reason charge is injected into the cell over several iterations. Such programming is slow and can cause errors since cells may be injected with extra unwanted charge. Other common errors in flash memory cells are due to charge leakage and reading disturbance that may cause charge to move from one cell to its adjacent cells. In order to overcome these problems, the novel framework of *rank modulation* was introduced in [8]. In this setup the information is carried by the relative ranking of the cells' charge levels and not by the absolute values of the charge levels. This allows for more efficient programming of cells, and coding by the ranking of the cells' levels is more robust to charge leakage than coding by their actual values. In this model codes are subsets of S_n, the set of all permutations on n elements, and the codewords are members of S_n, where each permutation corresponds to a ranking of n cells' levels from the highest one to the lowest. For example, the charge levels $(c_1, c_2, c_3, c_4) = (5, 1, 3, 4)$ are represented by the codeword $[1, 4, 3, 2]$ since the first cell has the highest level, the forth cell has the next highest level and so on.

To detect and/or correct errors caused by injection of extra charge or due to charge leakage we will use an appropriate distance measure. Several metrics on permutations are used for this purpose. In this paper we will consider only the Kendall's τ-metric [9], [10]. The Kendall's τ-distance between two permutation π_1 and π_2 in S_n is the minimum adjacent transpositions required to obtained π_2 from π_1, where adjacent transposition is an exchange of two distinct adjacent elements. For example, the Kendall's τ-distance between $\pi_1 = [2, 1, 4, 3]$ and $\pi_2 = [2, 4, 3, 1]$ is 2 as $[2, 1, 4, 3] \to [2, 4, 1, 3] \to [2, 4, 3, 1]$. Two permutations in this metric are at distance one if they differ in exactly one pair of adjacent elements. Distance one between these two permutations represents an exchange of two cells, which are adjacent in the permutation, due to a small changes in their charge level which changes their order.

Gray codes are very important in the context of rank modulation as was explained in [8]. They are used in many other applications, see [3], [12]. An excellent survey on Gray codes is given in [11]. The usage of Gray codes for rank modulation was also discussed in [5], [6], [8], and [13]. The permutations of S_n in the rank modulation scheme represent "new" logical levels of the flash memory. The codewords in the Gray code provide the order of these levels which should be implemented in various algorithms with the rank modulation scheme. Usually, a Gray code is just a simple cycle in a graph, in which the edges are defined between vertices with distance one in a given metric. Two adjacent vertices in the graph represent on one hand two elements whose distance is one by the given metric; and on the other hand a move from a vertex to a vertex implied by an operation defined by the metric. A snake-in-the-box code is a Gray code in which two elements in the code are not adjacent in the graph, unless they are consecutive in the code. Such a Gray code can detect a single error in a codeword. Snake-in-the-box codes were mainly discussed in the context of the Hamming scheme, e.g. [1].

In the rank modulation scheme the Gray code is defined slightly different since the operation is not defined by a metric. The permutation is defined by the order of the charge

Manuscript received February 26, 2014; revised July 13, 2014; accepted September 6, 2014. Date of publication October 1, 2014; date of current version October 16, 2014. This work was supported in part by the U.S.-Israel Binational Science Foundation, Jerusalem, Israel, under Grant 2012016. This paper was presented in part at the 2014 Information Theory and Applications Workshop.

The authors are with the Department of Computer Science, Technion-Israel Institute of Technology, Haifa 32000, Israel (e-mail: michalho@cs.technion.ac.il; etzion@cs.technion.ac.il).

Communicated by N. Kashyap, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2014.2359193

levels, from the highest one to the lowest one. From a given ranking of the charge levels, which defines a permutation, the next ranking is obtained by raising the charge level of one of the cells to be the highest level. This operation, called "push-to-the-top", is used in the rank modulation scheme. For example, the charge levels $(c_1, c_2, c_3, c_4) = (5, 1, 3, 4)$ are represented by the codeword $[1, 4, 3, 2]$, and by applying push-to-the-top operation on the second cell which has the lowest charge level, we have, for example, the charge levels $(c_1, c_2, c_3, c_4) = (5, 6, 3, 4)$ which are represented by the codeword $[2, 1, 4, 3]$. Hence, the permutation π_2 can follow the permutation π_1 if π_2 is obtained from π_1 by applying a push-to-the-top operation on π_1. Therefore, the related graph is directed with an outgoing edge from the vertex which represents π_1 into the vertex which represents π_2. On the other hand, one possible metric for the scheme is the Kendall's τ-metric. A Gray code (and a snake-in-the-box code as a special case) related to the rank modulation scheme is a directed simple cycle in the graph. In a snake-in-the-box code, related to this scheme, there is another requirement that the Kendall's τ-distance between any two codewords is at least two, including consecutive codewords. For example, $C = ([1, 2, 3, 4], [4, 1, 2, 3], [2, 4, 1, 3], [3, 2, 4, 1], [4, 3, 2, 1], [1, 4, 3, 2], [3, 1, 4, 2], [2, 3, 1, 4])$ is a snake-in-the-box code in S_4. The Kendall's τ-distance between any two permutations in C is at least 2.

One of the most important problems in the research on snake-in-the-box codes is to construct the largest possible code for the given graph. In a snake-in-the-box code for the rank modulation scheme we would like to find such a code with the largest number of permutations. In a recent paper by Yehezkeally and Schwartz [13], the authors constructed a snake-in-the-box code of length $M_{2n+1} = (2n + 1)(2n - 1)M_{2n-1}$ for permutations of S_{2n+1}, from a snake of length M_{2n-1} for permutations of S_{2n-1}. We will improve on this result by constructing a snake of length $M_{2n+1} = ((2n + 1)2n - 1)M_{2n-1}$ for permutations of S_{2n+1}, from a snake of length M_{2n-1} for permutations of S_{2n-1}. Thus, we have $\lim_{n\to\infty} \frac{M_{2n+1}}{S_{2n+1}} \approx 0.4338$, improving on the previous known ratio of $\lim_{n\to\infty} \frac{1}{\sqrt{\pi n}}$ [13]. For these constructions of snake-in-the-box codes we need an initial snake-in-the-box code and the largest one known to start both constructions is a snake of length 57 for permutations of S_5. We also propose a direct construction to form a snake of length $\frac{(2n+1)!}{2} - 2n + 1$ for permutations of S_{2n+1}. The direct construction was applied successfully for S_7 and S_9. This implies a better initial condition for the recursive constructions, and the ratio $\lim_{n\to\infty} \frac{M_{2n+1}}{S_{2n+1}} \approx 0.4743$.

The rest of this paper is organized as follows. In Section II we will define the basic concepts of Gray codes in the rank modulation scheme, the push-to-the-top operation, and the Kendall's τ-metric required in this paper. In Section III we present the main ideas and a framework for constructions of snake-in-the-box codes. In Section IV we present a recursive construction based on the given framework. This construction is used to obtain snake-in-the-box codes longer than the ones known before. In Section V, based on the framework, we present an idea for a direct construction based on necklaces. The construction is used to obtain snake-in-the-box codes of length $\frac{(2n+1)!}{2} - 2n + 1$ in S_{2n+1}, which we believe are optimal. The construction was applied successfully on S_7 and on S_9, and we conjecture that it can be applied on S_n for any odd $n > 6$. Conclusions and problems for future research are presented in Section VI.

II. Preliminaries

In this section we will repeat some notations defined and mentioned in [13], and we also present some other definitions.

Let $[n] \triangleq \{1, 2, \ldots, n\}$ and let $\pi = [a_1, a_2, \ldots, a_n]$ be a permutation over $[n]$, i.e., a permutation in S_n, such that for each $i \in [n]$ we have that $\pi(i) = a_i$.

Given a set $\mathcal{S}$ and a subset of transformations $T \subseteq \{f | f : \mathcal{S} \to \mathcal{S}\}$, a *Gray code* over $\mathcal{S}$ of size M, using transitions from T, is a sequence $C = (c_0, c_1, \ldots, c_{M-1})$ of M distinct elements from $\mathcal{S}$, called *codewords*, such that for each $j \in [M - 1]$ there exists a $t \in T$ for which $c_j = t(c_{j-1})$. The Gray code is called *complete* if $M = |\mathcal{S}|$, and *cyclic* if there exists $t \in T$ such that $c_0 = t(c_{M-1})$. Throughout this paper we will consider only cyclic Gray codes.

In the context of rank modulation for flash memories, $\mathcal{S} = S_n$ and the set of transformations T comprises of push-to-the-top operations. We denote by t_i the *push-to-the-top* operation on index i, $2 \le i \le n$, defined by

$$t_i([a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n]) = [a_i, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n].$$

and a *p-transition* will be an abbreviated notation for a push-to-the-top operation.

A sequence of p-transitions will be called a *transitions sequence*. A permutation π_0 and a transitions sequence $t_1, t_2, \ldots t_\ell$ define a sequence of permutations $\pi_0, \pi_1, \pi_2, \ldots, \pi_{\ell-1}, \pi_\ell$, where $\pi_i = t_i(\pi_{i-1})$, for each i, $1 \le i \le \ell$. This sequence is a cyclic Gray code, if $\pi_\ell = \pi_0$ and for each $0 \le i < j < \ell$, $\pi_i \ne \pi_j$. In the sequel the word cyclic will be omitted.

Given a permutation $\pi = [a_1, a_2, \ldots, a_n] \in S_n$, an *adjacent transposition* is an exchange of two distinct adjacent elements a_i, a_{i+1}, in π, for some $1 \le i \le n - 1$. The result of such an adjacent transposition is the permutation $[a_1, \ldots, a_{i-1}, a_{i+1}, a_i, a_{i+2}, \ldots, a_n]$. The *Kendall's τ-distance* [10] between two permutations $\pi_1, \pi_2 \in S_n$ denoted by $d_K(\pi_1, \pi_2)$ is the minimum number of adjacent transpositions required to obtain the permutation π_2 from the permutation π_1. A *snake-in-the-box code* is a Gray code in which for each two permutations π_1 and π_2 in the code we have $d_K(\pi_1, \pi_2) \ge 2$. Hence, a snake-in-the-box code is a Gray code capable of detecting one Kendall's τ-error. We will call such a snake-in-the-box code a *$\mathcal{K}$-snake*. We further denote by *$(n, M, \mathcal{K})$-snake* a $\mathcal{K}$-snake of size M with permutations from S_n. A $\mathcal{K}$-snake can be represented in two different equivalent ways:

- the sequence of codewords (permutations),
- the transitions sequence along with the first permutation.

Let $\mathcal{T}$ be a transitions sequence and let π be a permutation in S_n. If a $\mathcal{K}$-snake is obtained by applying $\mathcal{T}$ on π then a $\mathcal{K}$-snake will be obtained by using any other permutation from S_n instead of π. This is a simple observation from the fact that $t(\pi_2(\pi_1)) = \pi_2(t(\pi_1))$, where t is a p-transition and $\pi_2(\pi_1)$ refers to applying the permutation $\pi_2 \in S_n$ on the permutation $\pi_1 \in S_n$. In other words applying $\mathcal{T}$ on a different permutation just permutes the symbols, by a fixed given permutation, in all the resulting permutations when $\mathcal{T}$ is applied on π. Therefore, such a transitions sequence $\mathcal{T}$ will be called an *S-skeleton*.

For a transitions sequence $\sigma = t_{k_1}, t_{k_2}, \dots t_{k_\ell}$ and a permutation $\pi \in S_n$, we denote by $\sigma(\pi)$, the permutation obtained by applying the sequence of p-transitions in σ on π, i.e., t_{k_1} is applied on π, t_{k_2} is applied on $t_{k_1}(\pi)$, and so on. In other words, $\sigma(\pi) = (t_{k_1} \circ t_{k_2} \circ \dots \circ t_{k_\ell})(\pi) = t_{k_\ell}\left(t_{k_{l-1}}\left(\dots t_{k_2}\left(t_{k_1}(\pi)\right)\right)\right)$. Let σ_1, σ_2 be two transitions sequences. We say that σ_1 and σ_2 are *matching sequences*, and denote it by $\sigma_1 \leftrightsquigarrow \sigma_2$, if for each $\pi \in S_n$ we have $\sigma_1(\pi) = \sigma_2(\pi)$.

In [13] it was proved that a Gray code with permutations from S_n using only p-transitions on odd indices is a $\mathcal{K}$-snake. By starting with an even permutation and using only p-transitions on odd indices we get a sequence of even permutations, i.e., a subset of A_n, the alternating group of order n. This observation saves us the need to check whether a Gray code is in fact a $\mathcal{K}$-snake, at the cost of restricting the permutations in the $\mathcal{K}$-snake to the set of even permutations. However, the following assertions were also proved in [13].

- If C is an $(n, M, \mathcal{K})$-snake then $M \leq \frac{|S_n|}{2}$.
- If C is an $(n, M, \mathcal{K})$-snake which contains a p-transition on an even index then $M \leq \frac{|S_n|}{2} - \frac{1}{n-1}\binom{\lfloor n/2 \rfloor - 1}{2}$.

This motivates not to use p-transitions on even indices. Since we will use only p-transitions on odd indices, we will describe our constructions only for even permutations with odd length.

III. Framework for Constructions of $\mathcal{K}$-Snakes

In this section we present a framework for constructing $\mathcal{K}$-snakes in S_{2n+1}. Our snakes will contain only even permutations. We start by partitioning the set of even permutations of S_{2n+1} into classes. Next, we describe how to merge $\mathcal{K}$-snakes of different classes into one $\mathcal{K}$-snake. We conclude this section by describing how to combine most of these classes by using a hypergraph whose vertices represent the classes and whose edges represent the classes that can be merge together in one step.

We present two constructions for a $(2n+1, M_{2n+1}, \mathcal{K})$-snake, C_{2n+1}, one recursive and one direct. In this section we present the framework for these constructions. First, the permutations of A_{2n+1}, the set of even permutations from S_{2n+1}, are partitioned into classes, where each class induces one $\mathcal{K}$-snake which contains permutations only from the class. All these snakes have the same S-skeleton. Let L_{2n+1} be the set of all the classes.

The construction of C_{2n+1} from the $\mathcal{K}$-snakes of L_{2n+1} proceeds by a sequence of joins, where at each step we have a main $\mathcal{K}$-snake, and two $\mathcal{K}$-snakes from the remaining $\mathcal{K}$-snakes of L_{2n+1} are joined to the current main $\mathcal{K}$-snake. A join is performed by replacing one transition in the main $\mathcal{K}$-snake with a matching sequence.

In order to join the $\mathcal{K}$-snakes we need the following lemmas, for which the first can be easily verified. In the sequel, let $\sigma^k \triangleq \underbrace{\sigma \circ \sigma \circ \dots \circ \sigma}_{k\, times}$, i.e., performing the transitions sequence σ, k times.

Lemma 1: If $\alpha, \beta \in S_n$ then $\beta = t_i(\alpha)$ if and only if $\alpha = t_i^{i-1}(\beta)$.

Lemma 2: If $i \in [n-2]$ then $t_i \leftrightsquigarrow t_{i+2} \circ (t_i^{i-1} \circ t_{i+2})^2$.

Proof: Let $\alpha = [a_1, a_2, \dots, a_i, a_{i+1}, a_{i+2}, \dots, a_n]$ be a permutation over [n].

$$\begin{aligned}
&t_{i+2}(\alpha)\\
&\quad = [a_{i+2}, a_1, \dots, a_i, a_{i+1}, a_{i+3}, \dots, a_n],\\
&t_i^{i-1}(t_{i+2}(\alpha))\\
&\quad = [a_1, a_2, \dots, a_{i-1}, a_{i+2}, a_i, a_{i+1}, a_{i+3}, \dots, a_n],\\
&t_{i+2}(t_i^{i-1}(t_{i+2}(\alpha)))\\
&\quad = [a_{i+1}, a_1, a_2, \dots, a_{i-1}, a_{i+2}, a_i, a_{i+3}, \dots, a_n],\\
&t_i^{i-1}(t_{i+2}(t_i^{i-1}(t_{i+2}(\alpha))))\\
&\quad = [a_1, a_2, \dots, a_{i-1}, a_{i+1}, a_{i+2}, a_i, a_{i+3}, \dots, a_n],
\end{aligned}$$

and hence we have,

$$\begin{aligned}
&t_{i+2}(t_i^{i-1}(t_{i+2}(t_i^{i-1}(t_{i+2}(\alpha)))))\\
&\quad = [a_i, a_1, \dots, a_{i-1}, a_{i+1}, a_{i+2}, \dots, a_n]\\
&\quad = t_i(\alpha).
\end{aligned}$$

■

Corollary 1: If $\pi \in S_{2n+1}$ then $t_{2n-1}(\pi) = t_{2n+1}\left(t_{2n-1}^{2n-2}\left(t_{2n+1}\left(t_{2n-1}^{2n-2}\left(t_{2n+1}(\pi)\right)\right)\right)\right)$.

Lemma 2 can be generalized as follows (the following lemma is given for completeness, but it will not be used in the sequel, and hence its proof is omitted).

Lemma 3: If $i, j \in [n]$ and $|i - j| = k$, then $t_i \leftrightsquigarrow t_j \circ (t_i^{i-1} \circ t_j)^k$.

The partition of A_{2n+1} into the set of classes L_{2n+1} should satisfy the following properties:

(P1) The last two ordered elements of two permutations in the same class are equal.

(P2) Any two permutations which differ only by a cyclic shift of the first $2n-1$ elements, belong to the same class.

Corollary 2: Let π be a permutation in A_{2n+1}.

- *π and $t_{2n+1}(\pi)$ belong to different classes in L_{2n+1}.*
- *π and $t_{2n-1}(\pi)$ belong to the same class in L_{2n+1}.*

We continue now with the description of the method to join the $\mathcal{K}$-snakes of L_{2n+1} into C_{2n+1}. In the rest of the paper, A_{2n+1} is partitioned into classes according to the last two ordered elements in the permutations. Let $[x, y]$ denote the class of A_{2n+1} in which the last ordered pair in the permutations is (x, y). Let $\mathcal{T}$ be the S-skeleton of the $\mathcal{K}$-snakes in L_{2n+1}. Let $C_{\mathcal{T}}^{\pi}$ be a $\mathcal{K}$-snake for which $\mathcal{T}$ is its transitions sequence, and π is its first permutation. If π belongs to the class $[x, y]$, we say that $C_{\mathcal{T}}^{\pi}$ represents the class $[x, y]$. Note that all the permutations in $C_{\mathcal{T}}^{\pi}$ belong to the same class.

The transitions sequence $\mathcal{T}$ should satisfy the following properties (these properties are needed in order to make the required joins of cycles):

(P3) t_{2n-1} is the last transition in $\mathcal{T}$.

(P4) Given a permutation $\pi = [a_1, \dots, a_{2n}, a_{2n+1}]$, for each $x \in [2n+1] \setminus \{a_{2n}, a_{2n+1}\}$ there exists a permutation $\pi' \in C_{\mathcal{T}}^{\pi}$ whose last ordered three elements are (x, a_{2n}, a_{2n+1}).

Corollary 3: For each class $[x, y]$, *a permutation* $\pi \in [x, y]$, *and* $z \in [2n+1] \setminus \{x, y\}$, *there exists a permutation* $\pi' \in C_{\mathcal{T}}^{\pi}$ *whose last ordered three elements are* (z, x, y), *followed by the permutation* $t_{2n-1}(\pi')$.

Lemma 4: Let C *be a* $\mathcal{K}$*-snake which doesn't contain any permutation from the classes* $[y, z]$ *or* $[z, x]$, *let* $\pi = [a_1, a_2, \dots, a_{2n-2}, z, \mathbf{x}, \mathbf{y}]$ *be a permutation in* C *followed by* t_{2n-1}, *and let* σ *be a transitions sequence such that* $\mathcal{T} = \sigma \circ t_{2n-1}$. *Then replacing this* t_{2n-1} *transition in* C, *with*

$$t_{2n+1} \circ \sigma \circ t_{2n+1} \circ \sigma \circ t_{2n+1},$$

joins two $\mathcal{K}$*-snakes representing the classes* $[y, z]$ *and* $[z, x]$ *into* C *(after* π*).*

Proof: Observe that by Lemma 1 we have $\sigma \leftrightsquigarrow t_{2n-1}^{2n-2}$. Thus, we have

$$\begin{array}{rll}
\pi = & [a_1, a_2, \dots, a_{2n-2}, z, \mathbf{x}, \mathbf{y}] & \\
& \downarrow t_{2n+1} & \\
& \left.\begin{array}{l}[y, a_1, a_2, \dots, a_{2n-2}, \mathbf{z}, \mathbf{x}] \\ \downarrow \sigma \leftrightsquigarrow t_{2n-1}^{2n-2} \\ [a_1, a_2, \dots, a_{2n-2}, y, \mathbf{z}, \mathbf{x}]\end{array}\right\} & \begin{array}{l}\mathcal{K}-snake \\ for\ \ [z, x]\end{array} \\
& \downarrow t_{2n+1} & \\
& \left.\begin{array}{l}[x, a_1, a_2, \dots, a_{2n-2}, \mathbf{y}, \mathbf{z}] \\ \downarrow \sigma \leftrightsquigarrow t_{2n-1}^{2n-2} \\ [a_1, a_2, \dots, a_{2n-2}, x, \mathbf{y}, \mathbf{z}]\end{array}\right\} & \begin{array}{l}\mathcal{K}-snake \\ for\ \ [y, z]\end{array} \\
& \downarrow t_{2n+1} & \begin{array}{l}return\ to\ the \\ \mathcal{K}-snake\ C\end{array} \\
t_{2n-1}(\pi) = & [z, a_1, a_2, \dots, a_{2n-2}, \mathbf{x}, \mathbf{y}] &
\end{array}$$

■

The next step is to present an order for merging all the $\mathcal{K}$-snakes of L_{2n+1}, except one, into C_{2n+1}. This step will be performed by translating the merging problem into a 3-graph problem. We start with a sequence of definitions taken from [7].

Definition 5: A 3-graph (also called a 3-uniform hypergraph) $H = (V, E)$ *is a hypergraph where* V *is a set of vertices and* $E \subseteq \binom{V}{3}$. *A hyperedge of* H *will be called triple.*

A path in H *is an alternating sequence of* $\ell + 1$ *distinct vertices and* ℓ *distinct triples:* $v_0, e_1, v_1, \dots, v_{\ell-1}, e_\ell, v_\ell$, *with the property that* $\forall i \in [\ell] : v_{i-1}, v_i \in e_i$.

A cycle is a closed path, i.e. $v_0 = v_\ell$.

A sub-3-graph contains a subset $E' \subseteq E$ *and the subset* $V' \subseteq V$ *which contains all the vertices in* E'.

A tree T *in* H *is a connected sub-3-graph of* H *with no cycles.*

Let $H_{2n+1} = (V_{2n+1}, E_{2n+1})$ be a 3-graph defined as follows:

$$V_{2n+1} = \{[x, y] \ : \ x, y \in [2n+1], x \neq y\},$$
$$E_{2n+1} = \{\{[x, y], [y, z], [z, x]\} : x, y, z \in [2n+1],$$
$$x \neq y, x \neq z, y \neq z\}.$$

We denote a hyperedge $\{[x, y], [y, z], [z, x]\}$, where $x < y$ and $x < z$, by the triple $\langle x, y, z \rangle$.

The vertices in H_{2n+1} correspond to the classes in the set L_{2n+1}. Each $e \in E_{2n+1}$ contains three vertices, which correspond to three classes. These three classes can be represented by three $\mathcal{K}$-snakes, generated from the S-skeleton, which can be merged together by Corollary 3 and Lemma 4. Note that for any two edges e_1, e_2 in H_{2n+1} either $e_1 \cap e_2 = \varnothing$ or $|e_1 \cap e_2| = 1$. Let $T_{2n+1} = (V_{T_{2n+1}}, E_{T_{2n+1}})$ be a tree in H_{2n+1}. We join $|V_{T_{2n+1}}|$ $\mathcal{K}$-snakes which represent $|V_{T_{2n+1}}|$ classes of L_{2n+1} to form the $\mathcal{K}$-snake C_{2n+1}, by Corollary 3 and Lemma 4. The hyperedges which represent the joins which are performed are determined by T_{2n+1}, but these joins are not unique, and hence they can yield different final $\mathcal{K}$-snakes. The order in which the hyperedges are selected for these joins is also not unique, but this order doesn't affect the final $\mathcal{K}$-snakes. The size of the $\mathcal{K}$-snake C_{2n+1} depends on the number of vertices in the tree T_{2n+1}. A tree in a 3-graph contains an odd number of vertices [7]. Since in H_{2n+1} there are $(2n+1)(2n)$ vertices it follows that there is no tree in H_{2n+1} which contains all the vertices of V_{2n+1}. This motivates the following definition.

Definition 6: A nearly spanning tree in a 3-graph $H = (V, E)$ *is a tree in* H *which contains all the vertices of* V *except one.*

Now, let T_{2n+1} be a nearly spanning tree in H_{2n+1}.

Example 1: One choice for T_5 *is given below.*

The edges in the tree T_5 *are:*

$$\langle 1,2,5 \rangle, \quad \langle 1,2,4 \rangle, \quad \langle 1,2,3 \rangle, \quad \langle 1,4,5 \rangle,$$
$$\langle 2,5,4 \rangle, \quad \langle 1,3,4 \rangle, \quad \langle 2,4,3 \rangle, \quad \langle 1,5,3 \rangle, \quad \langle 2,3,5 \rangle.$$

The order of merging $\mathcal{K}$*-snakes from these classes obtained by this choice of* T_5 *can be chosen as follows.*

(1) *vertex* $[1, 2]$*;*
(2) *vertices* $[3, 1]$, $[2, 3]$, *(through the edge* $\langle 1, 2, 3 \rangle$*);*
(3) *vertices* $[4, 1]$, $[2, 4]$, *(through the edge* $\langle 1, 2, 4 \rangle$*);*
(4) *vertices* $[5, 1]$, $[2, 5]$, *(through the edge* $\langle 1, 2, 5 \rangle$*);*
(5) *vertices* $[5, 3]$, $[1, 5]$, *(through the edge* $\langle 1, 5, 3 \rangle$*);*
(6) *vertices* $[5, 2]$, $[3, 5]$, *(through the edge* $\langle 2, 3, 5 \rangle$*);*
(7) *vertices* $[3, 4]$, $[1, 3]$, *(through the edge* $\langle 1, 3, 4 \rangle$*);*
(8) *vertices* $[3, 2]$, $[4, 3]$, *(through the edge* $\langle 2, 4, 3 \rangle$*);*
(9) *vertices* $[4, 5]$, $[1, 4]$, *(through the edge* $\langle 1, 4, 5 \rangle$*);*
(10) *vertices* $[4, 2]$, $[5, 4]$, *(through the edge* $\langle 2, 5, 4 \rangle$*).*

Using the S-skeleton $\mathcal{T} = t_3, t_3, t_3$ *of the* $(3, 3, \mathcal{K})$*-snake, the snake-in-the-box code which is obtained by* T_5 *is a* $(5, 57, \mathcal{K})$*-snake presented in Figure 1. There is no* $(5, M, \mathcal{K})$*-snake for which* $M > 57$ *[13]. The S-skeleton of this code is* σ^3, *where*

$$\sigma = t_5, t_5, t_3, t_3, t_5, t_3, t_3, t_5, t_3, t_5, t_5, t_3, t_3, t_5, t_3, t_3, t_5, t_3, t_5$$

Theorem 7: If $n \geq 2$, *then there exists a nearly spanning tree* T_{2n+1} *in* H_{2n+1} *which doesn't include the vertex* $[2, 1]$.

Proof: We present a recursive construction for such a nearly spanning tree. We start with the nearly spanning tree given in Example 1. Note that T_5 doesn't include the vertex $[2, 1]$. Assume that there exists a nearly spanning tree, T_{2n-1}, in H_{2n-1}, which doesn't include the vertex $[2, 1]$. Note that H_{2n-1} is a sub-graph of H_{2n+1} and therefore T_{2n-1} is

3	2	1	3	2	5	3	2	4	3	1	5	3	1	2	3	1	4	3	5	2	1	5	2	4	5	2	3	5	1	4	5	1	2	5	1	3	5	4	2	1	4	2	3	4	2	5	4	1	3	4	1	2	4	1	5	4
4	3	2	1	3	2	5	3	2	4	3	1	5	3	1	2	3	1	4	3	5	2	1	5	2	4	5	2	3	5	1	4	5	1	2	5	1	3	5	4	2	1	4	2	3	4	2	5	4	1	3	4	1	2	4	1	5
5	4	3	2	1	3	2	5	3	2	4	3	1	5	3	1	2	3	1	4	3	5	2	1	5	2	4	5	2	3	5	1	4	5	1	2	5	1	3	5	4	2	1	4	2	3	4	2	5	4	1	3	4	1	2	4	1
1	5	4	4	4	1	1	1	5	5	2	4	4	4	5	5	5	2	2	1	4	3	3	3	1	1	1	4	4	2	3	3	3	4	4	4	2	2	1	3	5	5	5	1	1	1	3	3	2	5	5	5	3	3	3	2	2
2	1	5	5	5	4	4	4	1	1	5	2	2	2	4	4	4	5	5	2	1	4	4	4	3	3	3	1	1	4	2	2	2	3	3	3	4	4	2	1	3	3	3	5	5	5	1	1	3	2	2	2	5	5	5	3	3

Fig. 1. A $(5, 57, \mathcal{K})$-snake obtained by T_5.

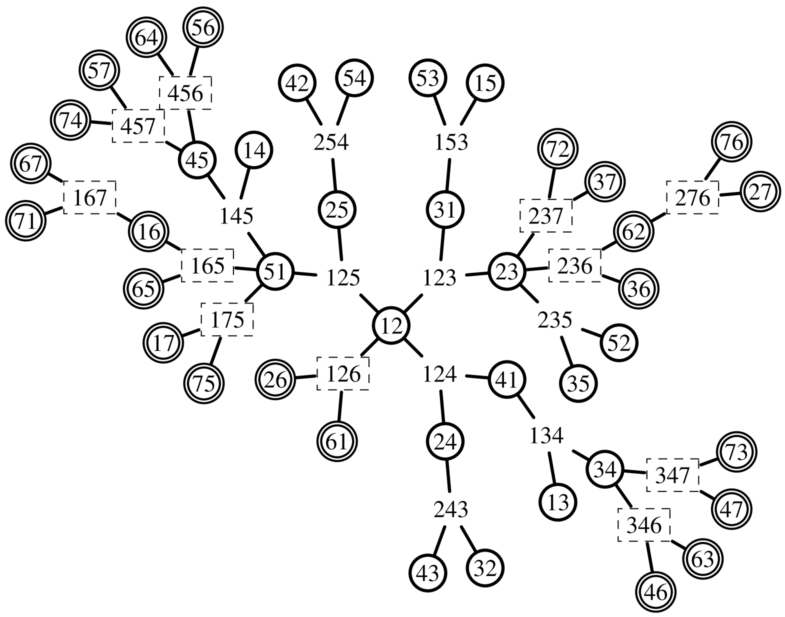

Fig. 2. The nearly spanning tree T_7 constructed from T_5.

a tree in H_{2n+1}. The vertices of H_{2n+1} which are not spanned by T_{2n-1} are

- $[x, 2n], [2n, x], [x, 2n + 1], [2n + 1, x]$ for each $x \in [2n - 1]$,
- $[2n, 2n + 1], [2n + 1, 2n]$,
- $[2, 1]$.

The nearly spanning tree T_{2n+1} is constructed from T_{2n-1} as follows. For each x, $2 \leq x \leq 2n - 2$, the edges $\langle x, x + 1, 2n\rangle$ and $\langle x, x + 1, 2n + 1\rangle$ are joined to T_{2n+1}; also the edges $\langle 1, 2, 2n\rangle$, $\langle 1, 2n, 2n - 1\rangle$, $\langle 1, 2n + 1, 2n - 1\rangle$, $\langle 1, 2n, 2n + 1\rangle$, and $\langle 2, 2n + 1, 2n\rangle$ are joined to T_{2n+1}. It is easy to verify that all the vertices of H_{2n+1} which are not spanned by T_{2n-1} (except for $[2, 1]$) are contained in the list of the edges which are joined to T_{2n-1}. When an edge is joined to the tree it has one vertex which is already in the tree and two vertices which are not on the tree. Hence, connectivity is preserved and no cycle is formed. Hence, it is easy to verify that by joining these edges to T_{2n-1} we form a nearly spanning tree in H_{2n+1}. ∎

Example 2: By using Theorem 7 and the nearly spanning tree T_5 of Example 1 we obtain the spanning tree T_7 depicted in Figure 2. The dashed boxes edges and the double lines nodes are added to T_5 in order to form T_7.

IV. A Recursive Construction

In this section we present the recursive construction for a $(2n + 1, M_{2n+1}, \mathcal{K})$-snake from a $(2n - 1, M_{2n-1}, \mathcal{K})$-snake. The construction is based on the nearly spanning tree T_{2n+1} presented in the previous section. Each of its vertices represent a class in which a $\mathcal{K}$-snake based on the $(2n - 1, M_{2n-1}, \mathcal{K})$-snake is generated. Those $\mathcal{K}$-snakes are merged together into one $(2n + 1, M_{2n+1}, \mathcal{K})$-snake using the framework presented in the previous section. We conclude this section with analyzing the length of the generated $\mathcal{K}$-snake compared the total number of permutations in S_{2n+1}.

We generate a $(2n + 1, M_{2n+1}, \mathcal{K})$-snake, C_{2n+1}, whose transitions sequence is $t_{k_1}, t_{k_2}, \ldots, t_{k_{M_{2n+1}}}$. C_{2n+1} has the following properties:

(Q1) k_j is odd for all $j \in [M_{2n+1}]$.
(Q2) $k_{M_{2n+1}} = 2n + 1$.
(Q3) For each $z \in [2n + 1]$ there exists a permutation $\pi \in C_{2n+1}$ such that $\pi(2n + 1) = z$.

The starting point of the recursive construction is $2n + 1 = 3$. The transitions sequence for $2n + 1 = 3$ is t_3, t_3, t_3, and the complete $(3, 3, \mathcal{K})$-snake is $C_3 \triangleq \{[1, 2, 3], [3, 1, 2], [2, 3, 1]\}$. Clearly (Q1), (Q2), and (Q3) hold for this transitions sequence and C_3.

Now, assume that there exists a $(2n - 1, M_{2n-1}, \mathcal{K})$-snake, C_{2n-1}, which satisfies properties (Q1), (Q2), (Q3), and let $\mathcal{T}_{2n-1} = t_{k_1}, t_{k_2}, \ldots, t_{k_{M_{2n-1}}}$ be its S-skeleton, i.e., $\mathcal{T}_{2n-1}$ is the transitions sequence of C_{2n-1}. Note that (Q1), (Q2), and (Q3) depend on the transitions sequence $\mathcal{T}_{2n-1}$ and are independent of the first permutation of C_{2n-1}. We construct a $(2n + 1, M_{2n+1}, \mathcal{K})$-snake, C_{2n+1}, where $M_{2n+1} = ((2n + 1)(2n) - 1)M_{2n-1}$, which also satisfies (Q1), (Q2), and (Q3).

First, all the permutations of A_{2n+1} are partitioned into $(2n+1)(2n)$ classes according to the last ordered two elements in the permutations. This implies that (P1) and (P2) are satisfied. In addition, (P3) and (P4) for $\mathcal{T}_{2n-1}$ are immediately implied by (Q2) and (Q3) for C_{2n-1}, respectively. Hence $\mathcal{T}_{2n-1}$ can be used as the S-skeleton for the $\mathcal{K}$-snakes in L_{2n+1}. Now, we merge the $\mathcal{K}$-snakes of the classes in L_{2n+1} (except [2, 1]), by using Lemma 4 and the nearly spanning tree T_{2n+1} of Theorem 7. We have to show that (Q1), (Q2), and (Q3) are satisfied for C_{2n+1}. (Q1) is readily verified. Clearly, t_{2n+1} was used to obtain C_{2n+1} (see Lemma 4), and therefore we can always define $\mathcal{T}_{2n+1}$ in such a way that its last transition is t_{2n+1}, and hence (Q2) is satisfied. For each $z \in [2n+1]$ there exists a class $[x, z]$ whose $\mathcal{K}$-snake is joined into C_{2n+1}, and therefore (Q3) is satisfied. Thus, we have

Theorem 8: Given a $(2n-1, M_{2n-1}, \mathcal{K})$-snake which satisfies (Q1), (Q2), and (Q3), we can obtain a $(2n+1, M_{2n+1}, \mathcal{K})$-snake, where $M_{2n+1} = ((2n+1)(2n)-1)M_{2n-1}$, which also satisfies (Q1), (Q2), and (Q3).

Following [13], we define $D_{2n+1} = \frac{M_{2n+1}}{(2n+1)!}$ as the ratio between the number of permutations in the given $(2n+1, M_{2n+1}, \mathcal{K})$-snake and the size of S_{2n+1}. Recall that if C is an $(2n+1, M, \mathcal{K})$-snake then $M \leq \frac{|S_{2n+1}|}{2}$, and we conjecture that the optimal size is $M = \frac{(2n+1)!}{2} - 2n + 1$. Thus, it is desirable to obtain a value D_{2n+1} close to half as much as possible. In our recursive construction $M_{2n+1} = ((2n+1)(2n)-1)M_{2n-1}$. Thus, we have

$$D_3 = \frac{1}{2},$$

$$\prod_{n=2}^{\infty} \frac{D_{2n+1}}{D_{2n-1}} = \frac{12\sqrt{\pi}}{5(1+\sqrt{5})\Gamma(\frac{1}{4}(5-\sqrt{5}))\Gamma(\frac{1}{4}(1+\sqrt{5}))},$$

which implies that

$$\lim_{n\to\infty} D_{2n+1} = \frac{1}{2} \cdot \frac{12\sqrt{\pi}}{5(1+\sqrt{5})\Gamma(\frac{1}{4}(5-\sqrt{5}))\Gamma(\frac{1}{4}(1+\sqrt{5}))} \approx 0.4338.$$

This computation can be done by any mathematical tool, e.g., WolframAlpha. This improves on the construction described in [13], which yields $M_{2n+1} = (2n+1)(2n-1)M_{2n-1}$ and $\lim_{n\to\infty} D_{2n+1} = \lim_{n\to\infty} \frac{1}{\sqrt{\pi n}}$.

V. A Direct Construction Based on Necklaces

In this section we describe a direct construction to form a $(2n+1, M_{2n+1}, \mathcal{K})$-snake. First, we describe a method to partition the classes which were used before into subclasses that are similar to necklaces. Next, we show how subclasses from different classes are merged into disjoint chains. Finally, we present a hypergraph and a graph in which we have to search for certain trees to form our desired $\mathcal{K}$-snake which we believe is of maximum length. Such $\mathcal{K}$-snakes were found in S_7 and S_9.

We present a direct construction for a $(2n+1, M_{2n+1}, \mathcal{K})$-snake, C_{2n+1}. The goal is to obtain $M_{2n+1} = \frac{(2n+1)!}{2} - (2n-1)$, and hence $\frac{D_{2n+1}}{D_{2n-1}} \geq 1 - \frac{1}{(2n)!}$. We believe that there is always a $(2n+1, M_{2n+1}, \mathcal{K})$-snake with $M_{2n+1} = \frac{(2n+1)!}{2} - (2n-1)$ and there is no such $\mathcal{K}$-snake with more codewords. We are making a slight change in the framework discussed in Section III. First, all the permutations of A_{2n+1} are partitioned into $(2n+1)(2n)$ classes according to the last ordered two elements. We denote by $[x, y]$ the class of all even permutations in which the last ordered pair in the permutation is (x, y). Each class is further partitioned into subclasses according to the cyclic order of the first $2n-1$ elements in the permutations, i.e., in each class $[x, y]$, the $\frac{(2n-1)!}{2}$ permutations are partitioned into $\frac{(2n-2)!}{2}$ disjoint subclasses. This implies that (P1) and (P2) are satisfied for both classes and subclasses. Let's denote each one of the subclasses by $[\alpha] - [x, y]$ where α is the cyclic order of the first $2n-1$ elements in the permutations of the subclass. Let α_1, α_2 be two permutations over $[2n+1] \setminus \{x, y\}$. If α_1 and α_2 have the same cyclic order, we denote it by $\alpha_1 \simeq \alpha_2$, otherwise $\alpha_1 \not\simeq \alpha_2$. Note that if $\alpha_1 \simeq \alpha_2$ then $[\alpha_1]-[x, y] = [\alpha_2]-[x, y]$. For example $[1, 2, 3] - [4, 5]$ represents the subclass with the permutations $[1, 2, 3, 4, 5]$, $[3, 1, 2, 4, 5]$, and $[2, 3, 1, 4, 5]$.

Let L_{2n+1} be the set of all classes, and let $\mathcal{T} = t_{2n-1}^{2n-1}$ be the S-skeleton of the $\mathcal{K}$-snakes in L_{2n+1}. Note that a $\mathcal{K}$-snake generated by $\mathcal{T}$ spans exactly all the permutations in one subclass. Hence (P3) and (P4) are immediately implied for both classes and subclasses. Such a $\mathcal{K}$-snake will be called a *necklace*. The slight change in the framework is that instead of one $\mathcal{K}$-snake, each class contains $\frac{(2n-2)!}{2}$ disjoint $\mathcal{K}$-snakes, all of them have the same S-skeleton.

The necklaces (subclasses) $[\alpha] - [x, y]$ are similar to necklaces on $2n-1$ elements. Joining the necklaces into one large $\mathcal{K}$-snake might be similar to the join of cycles from the pure cycling register of order $2n-1$, PCR_{2n-1}, into one cycle, which is also known as a de Bruijn sequence [2], [4]. There are two main differences between the two types of necklaces. The first one is that in de Bruijn sequences the necklaces do not represent permutations, but words of a given length over some finite alphabet. The second is that there is rather a simple mechanism to join all the necklaces into a de Bruijn sequence. We would like to have such a mechanism to join as many as possible necklaces from all the classes into one $\mathcal{K}$-snake.

Let T_{2n+1} be the nearly spanning tree constructed by Theorem 7. By repeated application of Lemma 4 according to the hyperedges of T_{2n+1} starting from a necklace in the class [1, 2] we obtain a $\mathcal{K}$-snake which contains exactly one necklace from each class $[x, y] \neq [2, 1]$. Such a $\mathcal{K}$-snake will be called a *chain*. If the chain contains the necklace $[\alpha]-[1, 2]$, we will denote it by $c[\alpha]$. For two permutations α_1 and α_2 over $[2n+1] \setminus \{1, 2\}$ such that $\alpha_1 \simeq \alpha_2$ we have $c[\alpha_1] = c[\alpha_2]$. Note that there is a unique way to merge the three necklaces which correspond to a hyperedge of T_{2n+1}, and hence there is no ambiguity in $c[\alpha]$ (even so the order of the joins is not unique), Note also that the transitions sequence of two distinct chains is usually different. The number of permutations in a chain is $((2n+1)(2n)-1)(2n-1)$. The following lemma is an immediate consequence of Lemma 4.

Lemma 9: Let $[x, y]$, $[y, z]$, and $[z, x]$ be three classes, and let α be a permutation of $[2n+1] \setminus \{x, y, z\}$. The necklaces $[\alpha, z]-[x, y]$, $[\alpha, y]-[z, x]$, and $[\alpha, x]-[y, z]$ can be merged together, where α, z is the sequence formed by concatenation of α and z.

Lemma 10: Let $[x, y]$, $[y, z]$, and $[z, x]$ be three classes. All the subclasses in these classes can be partitioned into disjoint sets, where each set contains exactly one necklace from each of the above three classes. The necklaces of each set can be merged together into one $\mathcal{K}$-snake.

Proof: For each permutation α over $[2n + 1] \setminus \{x, y, z\}$, the necklaces $[\alpha, z] - [x, y]$, $[\alpha, y] - [z, x]$, and $[\alpha, x] - [y, z]$ can be merged by Lemma 9. Thus, all the subclasses in these classes can be partitioned into disjoint sets. ∎

Corollary 4: The permutations of all the classes except for $[2, 1]$ can be partitioned into disjoint chains.

By Corollary 4 we construct $\frac{(2n-2)!}{2}$ disjoint chains which span A_{2n+1}, except for all the even permutations of the class $[2, 1]$. Recall that we have the same number, $\frac{(2n-2)!}{2}$, of $[2, 1]$-necklaces, which span all the permutations of the class $[2, 1]$. Now, we need a method to merge all these chains and necklaces, except for one necklace from the class $[2, 1]$, into one $\mathcal{K}$-snake C_{2n+1}. Note that for $2n + 1 = 5$ we have only one chain. Thus, this chain is the final $\mathcal{K}$-snake C_5. This $\mathcal{K}$-snake is exactly the same $\mathcal{K}$-snake as the one generated by the recursive construction in Section IV.

Lemma 11: Let x be an integer such that $3 \le x \le 2n + 1$, let α be a permutation of $[2n + 1] \setminus \{x, 2, 1\}$, and assume that the permutations $[\alpha, 1, x, 2]$ and $[\alpha, 2, 1, x]$ are contained in two distinct chains. We can merge these two chains via the necklace $[\alpha, x] - [2, 1]$.

Proof: Let c_1 be the chain which contains the permutation $\pi_1 = [\alpha, 1, x, 2]$, c_2 be the chain which contains the permutation $\pi_2 = [\alpha, 2, 1, x]$, and η be the necklace which contains the permutation $\pi_3 = [\alpha, x, 2, 1]$. Note that all the chains contains only the p-transitions t_{2n+1} and t_{2n-1}. The permutation $t_{2n+1}(\pi_1)$ appears in c_2, the permutation $t_{2n+1}(\pi_2)$ appears in η, and the permutation $t_{2n+1}(\pi_3)$ appears in c_1. Therefore, π_1, π_2, and π_3 are followed by t_{2n-1} in c_1, c_2, and η, respectively. Let σ_i, $i \in \{1, 2\}$, be a transitions sequence such that σ_i, t_{2n-1} is the transitions sequence of c_i, and therefore $t_{2n-1}(\sigma_i(\pi_i)) = \pi_i$. By Lemma 1 we have $\sigma_1 \rightsquigarrow t_{2n-1}^{2n-2} \leftrightsquigarrow \sigma_2$. Similarly to Lemma 4, by replacing the transition t_{2n-1} which follows π_3 in η, with $t_{2n+1} \circ \sigma_1 \circ t_{2n+1} \circ \sigma_2 \circ t_{2n+1}$, we merge c_1, c_2 and η into a $\mathcal{K}$-snake. Thus, we have

$$\begin{aligned}
\pi_3 &= [a_1, a_2, \ldots, a_{2n-2}, x, 2, 1] \\
&\quad \downarrow t_{2n+1} \\
&\quad [1, a_1, a_2, \ldots, a_{2n-2}, x, 2] \\
&\quad \downarrow \sigma_1 \leftrightsquigarrow t_{2n-1}^{2n-2} \qquad\qquad \textit{the chain } c_1 \\
\pi_1 &= [a_1, a_2, \ldots, a_{2n-2}, 1, x, 2] \\
&\quad \downarrow t_{2n+1} \\
&\quad [2, a_1, a_2, \ldots, a_{2n-2}, 1, x] \\
&\quad \downarrow \sigma_2 \leftrightsquigarrow t_{2n-1}^{2n-2} \qquad\qquad \textit{the chain } c_2 \\
\pi_2 &= [a_1, a_2, \ldots, a_{2n-2}, 2, 1, x] \\
&\quad \downarrow t_{2n+1} \qquad\qquad \textit{return to the necklace } \eta \\
t_{2n-1}(\pi_3) &= [x, a_1, a_2, \ldots, a_{2n-2}, 2, 1]
\end{aligned}$$

∎

For each x, $3 \le x \le 2n + 1$, and for each permutation α of $[2n + 1] \setminus \{x, 1, 2\}$, the merging of two distinct chains which contain the permutations $[\alpha, 1, x, 2]$ and $[\alpha, 2, 1, x]$ via the necklace $[\alpha, x] - [2, 1]$ as described in Lemma 11, will be denoted by $M[x]$-connection. Note that if $x \in \{3, 4, 5\}$ then the permutations $[\alpha, 1, x, 2]$ and $[\alpha, 2, 1, x]$ are contained in the same chain. Thus, there are no $M[3]$-connections, $M[4]$-connections, or $M[5]$-connections.

Lemma 11 suggests a method to join all the chains and all the $[2, 1]$-necklaces except one into a $\mathcal{K}$-snake of length $\frac{(2n+1)!}{2} - (2n - 1)$. This should be implemented by $\frac{(2n-2)!}{2} - 1$ iterations of the merging suggested by Lemma 11. The current merging problem is also translated into a $3 - graph$ problem (see Definition 5). Let $\hat{H}_{2n+1} = (\hat{V}_{2n+1}, \hat{E}_{2n+1})$ be a 3-graph defined as follows.

$$\begin{aligned}
\hat{V}_{2n+1} &= \{c[\alpha] \;:\; \alpha \text{ is a permutation of } [2n + 1] \setminus \{1, 2\}\} \\
&\quad \cup \{[\beta] - [2, 1] \;: \\
&\qquad\quad \beta \text{ is a permutation of } [2n + 1] \setminus \{1, 2\}\} \\
\hat{E}_{2n+1} &= \{\{c[\alpha_1], c[\alpha_2], [\beta] - [2, 1]\} \;: \\
&\quad c[\alpha_1] \text{ and } c[\alpha_2] \text{ can be merged together} \\
&\quad \text{via } [\beta] - [2, 1] \text{ by Lemma 11}\}.
\end{aligned}$$

The vertices in $\hat{V}_{2n+1}$ are of two types, chains and $[2, 1]$-necklaces. Each $e \in \hat{E}_{2n+1}$ contains three vertices, two chains and one necklace, which can be merged together by Lemma 11. Therefore, the edge will be signed by $M[x]$ as described before. Note that $\hat{E}_{2n+1}$ might contains parallel edges with different signs.

Let $\hat{T}_{2n+1} = (V_{\hat{T}_{2n+1}}, E_{\hat{T}_{2n+1}})$ be a nearly spanning tree in $\hat{H}_{2n+1}$. Note that such a nearly spanning tree must contain all the vertices in $\hat{V}_{2n+1}$ except for one $[2, 1]$-necklace. If such a nearly spanning tree exists then by Lemma 11, we can merge all the chains via $[2, 1]$-necklaces to form the $\mathcal{K}$-snake C_{2n+1}. This $\mathcal{K}$-snake contains all the permutations of A_{2n+1} except for $2n - 1$ permutations which form one $[2, 1]$-necklace.

The joins which are performed are determined by the edges of $\hat{T}_{2n+1}$. Note that there is a unique way to merge the three vertices which correspond to a hyperedge of $\hat{T}_{2n+1}$ signed by $M[x]$. Hence, by using the given spanning trees T_{2n+1} and $\hat{T}_{2n+1}$, there is no ambiguity in C_{2n+1} (even so the orders of the joins are not unique). However, different nearly spanning trees can yield different final $\mathcal{K}$-snakes. Note that the $\mathcal{K}$-snake C_{2n+1} generated by this construction has only t_{2n+1} and t_{2n-1} p-transitions, where usually t_{2n-1} is used. The p-transition t_{2n-1} is the only transition in the $\mathcal{K}$-snake of the subclasses. On average 3 out of $4n$ sequential p-transitions of C_{2n+1} are the p-transition t_{2n+1}. A similar property exists when a de Bruijn sequence is generated from the necklaces of pure cycling register of order n [2], [4].

Finding a nearly spanning tree $\hat{T}_{2n+1}$ is an open question. But, we found such trees for $n = 3$ and $n = 4$. We believe that a similar construction to the one which follows in the sequel for $n = 3$ and $n = 4$, exists for all $n > 4$.

Conjecture 1: For each $n \ge 2$, there exists a $(2n + 1, M_{2n+1}, \mathcal{K})$-snake, where $M_{2n+1} = \frac{(2n+1)!}{2} - (2n - 1)$ in which there are only t_{2n-1} and t_{2n+1} p-transitions.

Example 3: For $n = 3$, a $(7, 2515, \mathcal{K})$-snake is constructed by using the tree T_7 of Example 2, and the tree $\hat{T}_7$

defined below. $\hat{T}_7$ *contains* 12 *chains, where each chain contains* 41 *necklaces. It also contains* 11 $[2,1]$*-necklaces and* 11 *hyperedges. Denote an edge in* $\hat{H}_7$ *by* $(\{c_i, c_j, \eta_k\}, x)$ *where* $M[x]$ *is the sign of the edge.* $\hat{T}_7$ *is defined as follows.*

<u>*The chains in* $\hat{T}_7$:</u>

$c_1 = [3,4,5,6,7]-[1,2]$, $c_2 = [3,4,6,7,5]-[1,2]$,
$c_3 = [3,4,7,5,6]-[1,2]$, $c_4 = [3,5,4,7,6]-[1,2]$,
$c_5 = [3,5,6,4,7]-[1,2]$, $c_6 = [3,5,7,6,4]-[1,2]$,
$c_7 = [3,6,4,5,7]-[1,2]$, $c_8 = [3,6,5,7,4]-[1,2]$,
$c_9 = [3,6,7,4,5]-[1,2]$, $c_{10} = [3,7,4,6,5]-[1,2]$,
$c_{11} = [3,7,5,4,6]-[1,2]$, $c_{12} = [3,7,6,5,4]-[1,2]$.

<u>*The necklaces in* $\hat{T}_7$:</u>

$\eta_1 = [3,4,5,7,6]-[2,1]$, $\eta_2 = [3,4,6,5,7]-[2,1]$,
$\eta_3 = [3,4,7,6,5]-[2,1]$, $\eta_4 = [3,5,4,6,7]-[2,1]$,
$\eta_5 = [3,5,6,7,4]-[2,1]$, $\eta_6 = [3,5,7,4,6]-[2,1]$,
$\eta_7 = [3,6,4,7,5]-[2,1]$, $\eta_8 = [3,6,5,4,7]-[2,1]$,
$\eta_9 = [3,6,7,5,4]-[2,1]$, $\eta_{10} = [3,7,4,5,6]-[2,1]$,
$\eta_{11} = [3,7,5,6,4]-[2,1]$.

<u>*The edges in* $\hat{T}_7$:</u>

$e_1 = (\{c_{11}, c_6, \eta_9\}, 6)$, $e_2 = (\{c_6, c_1, \eta_2\}, 6)$,
$e_3 = (\{c_2, c_{12}, \eta_{11}\}, 6)$, $e_4 = (\{c_{12}, c_7, \eta_4\}, 6)$,
$e_5 = (\{c_5, c_3, \eta_3\}, 6)$, $e_6 = (\{c_3, c_4, \eta_7\}, 6)$,
$e_7 = (\{c_9, c_{10}, \eta_{10}\}, 6)$, $e_8 = (\{c_{10}, c_8, \eta_5\}, 6)$,
$e_9 = (\{c_{12}, c_9, \eta_8\}, 7)$, $e_{10} = (\{c_9, c_3, \eta_1\}, 7)$,
$e_{11} = (\{c_2, c_{11}, \eta_6\}, 7)$.

$\hat{H}_7$ *contains another* $[2,1]$*-necklace,* $\eta_{12} = [3,7,6,4,5]-[2,1]$*, and the following additional edges:*

$e_{12} = (\{c_1, c_{11}, \eta_{12}\}, 6)$, $e_{13} = (\{c_7, c_2, \eta_1\}, 6)$,
$e_{14} = (\{c_4, c_5, \eta_8\}, 6)$, $e_{15} = (\{c_8, c_9, \eta_6\}, 6)$,
$e_{16} = (\{c_{10}, c_2, \eta_2\}, 7)$, $e_{17} = (\{c_8, c_1, \eta_3\}, 7)$,
$e_{18} = (\{c_{11}, c_{10}, \eta_4\}, 7)$, $e_{19} = (\{c_3, c_{12}, \eta_5\}, 7)$,
$e_{20} = (\{c_6, c_7, \eta_7\}, 7)$, $e_{21} = (\{c_4, c_8, \eta_9\}, 7)$,
$e_{22} = (\{c_1, c_4, \eta_{10}\}, 7)$, $e_{23} = (\{c_5, c_6, \eta_{11}\}, 7)$,
$e_{24} = (\{c_7, c_5, \eta_{12}\}, 7)$.

An additional different illustration of $\hat{H}_7$ *is presented in the sequel (see Example 4).*

For each $n \geq 3$, let $\mathcal{G}_{2n+1} = (\mathcal{V}_{2n+1}, \mathcal{E}_{2n+1})$ be a multi-graph (with parallel edges) with labels and signs on the edges. The vertices of $\mathcal{V}_{2n+1}$ represent the $\frac{(2n-2)!}{2}$ chains and hence $|\mathcal{V}_{2n+1}| = \frac{(2n-2)!}{2}$. There is an edge signed with $M[x]$, where $6 \leq x \leq 2n+1$, between the vertex (chain) c_1 and vertex (chain) c_2, if c_1 contains a permutation $[\alpha, 2, 1, x]$ and c_2 contains the permutation $[\alpha, 1, x, 2]$, where $c_1 \neq c_2$. The label on this edge is the necklace $[\alpha, x]-[2,1]$. Note that the label on the edge is a necklace which can merge together the chains of its corresponding endpoints by $M[x]$-connection. Note also that the pair α, x might not be unique and hence the graph might have parallel edges. A spanning tree in $\mathcal{G}_{2n+1}$ which doesn't have two edges with the same label, will be called *a chain tree*. The following Lemma can be easily verified.

Lemma 12: There exists a nearly spanning tree in $\hat{H}_{2n+1}$ *if and only if there exists a chain tree in* $\mathcal{G}_{2n+1}$.

Henceforth, T_{2n+1} will be the nearly spanning tree constructed in Theorem 7, and the chains are constructed via T_{2n+1}.

Definition 13: Let $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ *and* $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2)$ *be two multi-graphs with labels and signs on the edges, where the set of the labels of* $\mathcal{G}_i$ *denoted by* $\mathcal{L}_i$, $i \in \{1,2\}$*. We say that* $\mathcal{G}_1$ *is isomorphic to* $\mathcal{G}_2$ *if there exist two bijective functions* $f: \mathcal{V}_1 \to \mathcal{V}_2$ *and* $g: \mathcal{L}_1 \to \mathcal{L}_2$*, with the following property:* $(u, v) \in \mathcal{E}_1$ *with the label* η *and sign* $M[x]$*, if and only if* $(f(u), f(u)) \in \mathcal{E}_2$ *with the label* $g(\eta)$ *and sign* $M[x]$.

Definition 14: For each $n \geq 4$*, a sub-graph of* $\mathcal{G}_{2n+1}$ *which is isomorphic to* $\mathcal{G}_{2n-1}$ *is called a component of* $\mathcal{G}_{2n+1}$*, and denoted by* $A = (\mathcal{V}_A, \mathcal{L}_A)$ *where* $\mathcal{V}_A$ *consists of the vertices (chains) of the component,* $\mathcal{L}_A$ *consists of the labels (*$[2,1]$*-necklaces) on the edges in the component. Note that* $|\mathcal{V}_A| = |\mathcal{L}_A|$*, i.e., the numbers of the distinct labels is equal to the number of the vertices.*

Definition 15: Two components, $A = (\mathcal{V}_A, \mathcal{L}_A)$ *and* $B = (\mathcal{V}_B, \mathcal{L}_B)$*, in* $\mathcal{G}_{2n+1}$ *are called disjoint if* $\mathcal{V}_A \cap \mathcal{V}_B = \varnothing$ *and* $\mathcal{L}_A \cap \mathcal{L}_B = \varnothing$*, i.e., there is no a common vertex (chain) or a common label (*$[2,1]$*-necklace) in* A *and* B.

Lemma 16: For each $n \geq 4$, $\mathcal{G}_{2n+1}$ *consists of* $(2n-3)(2n-2)$ *disjoint copies of isomorphic graphs to* $\mathcal{G}_{2n-1}$*, called components. The edges between the vertices of two distinct components are signed only with* $M[2n]$ *and* $M[2n+1]$.

Proof: The $M[x]$-connections are deduced by the tree T_{2n+1}, which was used for the construction of the chains. In particular, the path between the vertices $[1,x]$ and $[x,2]$ in T_{2n+1} determines the $M[x]$-connections in $\mathcal{G}_{2n+1}$. By Theorem 7, T_{2n-1} is a sub-graph of T_{2n+1}. Therefore, for each x, $x \geq 3$, the path between the vertices $[1,x]$ and $[x,2]$ in T_{2n+1} is equal to the path between the vertices $[1,x]$ and $[x,2]$ in T_{2k+1} for each $x \leq 2k+1 \leq 2n+1$. The number of the vertices (chains) in $\mathcal{G}_{2n+1}$ is equal to $\frac{(2n-2)!}{2}$, and each component contains $\frac{(2n-4)!}{2}$ vertices. Thus, $\mathcal{G}_{2n+1}$ consists of $(2n-3)(2n-2)$ disjoint copies of isomorphic graphs to $\mathcal{G}_{2n-1}$ connected by edges signed only with $M[2n]$ and $M[2n+1]$. ■

For each $n \geq 4$, let $\hat{\mathcal{G}}_{2n+1} = (\hat{\mathcal{V}}_{2n+1}, \hat{\mathcal{E}}_{2n+1})$ be the *component graph* of $\mathcal{G}_{2n+1}$. The vertices of $\hat{\mathcal{V}}_{2n+1}$ represent the components of $\mathcal{G}_{2n+1}$, There is an edge signed with $M[x]$, $x \in \{2n, 2n+1\}$, between the vertices (components) A and B, if the chain that contains the permutation $[\alpha, 2, 1, x]$ is contained in A, and the chain that contains the permutation $[\alpha, 1, x, 2]$ is contained in B. The label on this edge is the necklace $[\alpha, x]-[2,1]$. We define $\hat{\mathcal{G}}_7$ to be $\mathcal{G}_7$, i.e., each component of $\hat{\mathcal{G}}_7$ consists of exactly one chain (and also one distinct $[2,1]$-necklace in order to follow the properties of $\hat{\mathcal{G}}_{2n+1}$).

Definition 17: A components spanning tree, $\hat{T}_{2n+1}$ *is a spanning tree in* $\hat{\mathcal{G}}_{2n+1}$*, where in the set of the labels of the tree's edges, there are no two labels from the same component, i.e., each label in the set of the labels of the tree's edges belongs to a different component.*

Example 4: $\hat{\mathcal{G}}_7$ *is depicted in Figure 3, where the vertices numbers and the edges labels corresponds to the chains and the necklaces in Example 3, respectively. The vertical edges are signed with* $M[6]$*, while the horizontal edges are signed with* $M[7]$*. The double lines edges correspond to the edges of* $\hat{T}_7$.

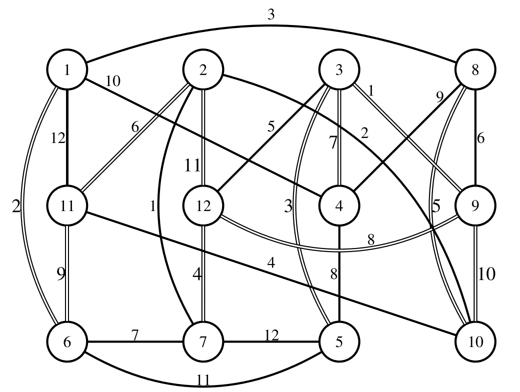

Fig. 3. The graph $\hat{\mathcal{G}}_7$ and its component spanning tree $\hat{T}_7$.

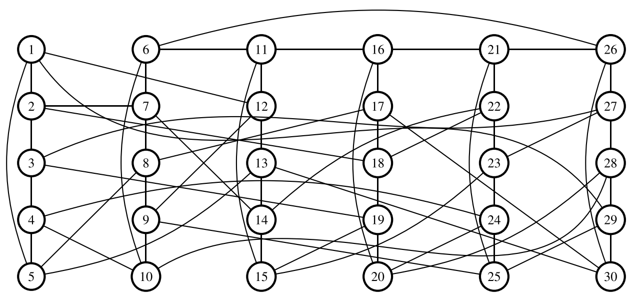

Fig. 4. The graph $\hat{\mathcal{G}}_9$.

Conjecture 2: For each component A in $\hat{\mathcal{G}}_{2n+1}$, $n \geq 3$, and for each label η of A, there exists a components spanning tree, where there is no edge in the tree with the label η.

Conjecture 2 implies Conjecture 1, i.e.,

Theorem 18: If Conjecture 2 is true then for each $n \geq 2$, there exists a $(2n+1, M_{2n+1}, \mathcal{K})$-snake, where $M_{2n+1} = \frac{(2n+1)!}{2} - (2n-1)$ in which there are only t_{2n-1} and t_{2n+1} p-transitions.

Conjecture 2 was verified by computer search for $n = 3$ and $n = 4$. By using Conjecture 2 recursively, for each $n \geq 3$, and for each necklace η in class $[2, 1]$, we can construct a chain tree T in $\mathcal{G}_{2n+1}$, which doesn't include η as a label on an edge in T.

Corollary 5: There exist a $(7, 2515, \mathcal{K})$-snake and a $(9, 181433, \mathcal{K})$-snake, and hence $\lim_{n\to\infty} \frac{M_{2n+1}}{S_{2n+1}} \approx 0.4743$.

Note that the ratio $\lim_{n\to\infty} \frac{M_{2n+1}}{S_{2n+1}}$ would be improved, if there exists a $(2m+1, \frac{(2m+1)!}{2} - (2m-1), \mathcal{K})$-snake for some $m > 4$.

Conjecture 3: The $(2n-3)(2n-2)$ components in $\hat{\mathcal{G}}_{2n+1}$ can be arranged in a $(2n-3) \times (2n-2)$ grid. The edges which are sign by $M[2n]$ define $2n-2$ cycles of length $2n-3$. Each cycle contains the vertices of exactly one column, and is called an $M[2n]$-cycle. The edges which are sign with $M[2n+1]$ are between two components in different columns, and they also define $2n-2$ cycles of length $2n-3$. Such a cycle will be called an $M[2n+1]$-cycle. Each multi-edge between two components has $\frac{(2n-4)!}{2}$ parallel edges (the number of chains in the component). Parallel edges have the same sign x, $x \in \{2n, 2n+1\}$, but different labels (i.e., $M[x]$-connection, but with different $[2, 1]$-necklaces).

Example 5: An illustration for the structure of $\hat{\mathcal{G}}_{2n+1}$ for $n = 3$ is presented in Example 4, and for $n = 4$ is depicted in Figure 4. In $\hat{\mathcal{G}}_9$ there are 30 components, where each component is isomorphic to $\hat{\mathcal{G}}_7$ (thus, it contains 12 chains and 12 $[2, 1]$-necklaces).

VI. Conclusions and Future Research

Gray codes for permutations using the operation push-to-the-top and the Kendall's τ-metric were discussed. We have presented a framework for constructing snake-in-the-box codes for S_n. The framework for the construction yield a recursive construction with large snakes. A direct construction to obtain snakes which might be optimal in length was also presented. Several questions arise from our discussion and they are considered for current and future research.

1) Complete the direct construction for snakes of length $\frac{(2n+1)!}{2} - 2n + 1$ in S_{2n+1}.
2) Can a snake in S_{2n+1} have size larger than $\frac{(2n+1)!}{2} - 2n + 1$?
3) Prove or disprove that the length of the longest snake in S_{2n} is not longer than the length of the longest snake in S_{2n-1}.
4) Examine the questions in this paper for the ℓ_∞ metric.

Acknowledgment

The authors would like to thank the anonymous reviewers for their careful reading of the paper. Especially, one of the reviewers pointed out on the good ratio $\lim_{n\to\infty} \frac{M_{2n+1}}{S_{2n+1}} \approx 0.4338$ compared to one in [13].

References

[1] H. L. Abbott and M. Katchalski, "On the snake in the box problem," *J. Combinatorial Theory B*, vol. 45, no. 1, pp. 13–24, 1988.

[2] T. Etzion and A. Lempel, "Algorithms for the generation of full-length shift-register sequences," *IEEE Trans. Inf. Theory*, vol. 30, no. 3, pp. 480–484, May 1984.

[3] T. Etzion and K. G. Paterson, "Near optimal single-track Gray codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 779–789, May 1996.

[4] H. Fredricksen, "A survey of full length nonlinear shift register cycle algorithms," *SIAM Rev.*, vol. 24, no. 2, pp. 195–221, Apr. 1982.

[5] E. En Gad, M. Langberg, M. Schwartz, and J. Bruck, "Constant-weight Gray codes for local rank modulation," *IEEE Trans. Inf. Theory*, vol. 57, no. 11, pp. 7431–7442, Nov. 2011.

[6] E. En Gad, M. Langberg, M. Schwartz, and J. Bruck, "Generalized Gray codes for local rank modulation," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6664–6673, Oct. 2013.

[7] A. Goodall and A. de Mier, "Spanning trees of 3-uniform hypergraphs," *Adv. Appl. Math.*, vol. 47, no. 4, pp. 840–868, 2011.

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

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

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

[11] C. Savage, "A survey of combinatorial Gray codes," *SIAM Rev.*, vol. 39, no. 4, pp. 605–629, 1997.

[12] C. D. Savage and P. Winkler, "Monotone Gray codes and the middle levels problem," *J. Combinatorial Theory A*, vol. 70, no. 2, pp. 230–248, 1995.

[13] Y. Yehezkeally and M. Schwartz, "Snake-in-the-box codes for rank modulation," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5471–5483, Aug. 2012.

Michal Horovitz was born in Israel in 1987. She received the B.Sc. degree from the Open University of Israel, Ra'anana, Israel, in 2009, from the department of Mathematics and from the department of Computer Science. She is currently a Ph.D. student in the Computer Science Department at the Technion - Israel Institute of Technology, Haifa, Israel.

Her research interests include coding theory with applications to non-volatile memories and combinatorics.

Tuvi Etzion (M'89–SM'94–F'04) was born in Tel Aviv, Israel, in 1956. He received the B.A., M.Sc., and D.Sc. degrees from the Technion - Israel Institute of Technology, Haifa, Israel, in 1980, 1982, and 1984, respectively.

From 1984 he held a position in the Department of Computer Science at the Technion, where he has a Professor position. During the years 1985-1987 he was Visiting Research Professor with the Department of Electrical Engineering - Systems at the University of Southern California, Los Angeles. During the summers of 1990 and 1991 he was visiting Bellcore in Morristown, New Jersey. During the years 1994-1996 he was a Visiting Research Fellow in the Computer Science Department at Royal Holloway College, Egham, England. He also had several visits to the Coordinated Science Laboratory at University of Illinois in Urbana-Champaign during the years 1995-1998, two visits to HP Bristol during the summers of 1996, 2000, a few visits to the Department of Electrical Engineering, University of California at San Diego during the years 2000-2012, and several visits to the Mathematics Department at Royal Holloway College, Egham, England, during the years 2007-2009.

His research interests include applications of discrete mathematics to problems in computer science and information theory, coding theory, and combinatorial designs.

Dr Etzion was an Associate Editor for Coding Theory for the IEEE TRANSACTIONS ON INFORMATION THEORY from 2006 till 2009. From 2004 to 2009, he was an Editor for the *Journal of Combinatorial Designs*. From 2011 he is an Editor for *Designs, Codes, and Cryptography*. From 2013 he is an Editor for *Advances of Mathematics in Communications*.