

# More restricted growth functions: Gray codes and exhaustive generations

Ahmad Sabri<sup>1</sup> and Vincent Vajnovszki<sup>2</sup>

<sup>1</sup>Center for Computational Mathematics Studies, Gunadarma University  
 sabri@staff.gunadarma.ac.id

<sup>2</sup>LE2I, Université Bourgogne Franche-Comté  
 vvajnov@u-bourgogne.fr

March 20, 2017

## Abstract

A Gray code for a combinatorial class is a method for listing the objects in the class so that successive objects differ in some prespecified, small way, typically expressed as a bounded Hamming distance. In a previous work, the authors of the present paper showed, among other things, that the  $m$ -ary Reflected Gray Code Order yields a Gray code for the set of restricted growth functions. Here we further investigate variations of this order relation, and give the first Gray codes and efficient generating algorithms for bounded restricted growth functions.

**Keywords:** *Gray code (order), restricted growth function, generating algorithm*

## 1 Introduction

In [4] the authors shown that both the order relation induced by the generalization of the Binary Reflected Gray Code and one of its suffix partitioned version yield Gray codes on some sets of restricted integer sequences, and in particular for restricted growth functions. These results are presented in a general framework, where the restrictions are defined by means of statistics on integer sequences.

In the present paper we investigate two prefix partitioning order relations on the set of *bounded* restricted growth functions: as in [4], the original Reflected Gray Code Order on  $m$ -ary sequences, and a new order relation which is an appropriate modification of the former one. We show that, according to the parity of the imposed bound, one of these order relations gives a Gray code on the set of bounded restricted growth functions. As a byproduct, we obtain a Gray code for restricted growth functions with a specified odd value for the largest entry; the case of an even value of the largest entry remains an open problem. In the final part we present the corresponding exhaustive generating algorithms. A preliminary version of these results were presented at The Japanese Conference on Combinatorics and its Applications in May 2016 in Kyoto [5].

## 2 Notation and definitions

A *restricted growth function* of length  $n$  is an integer sequence  $\mathbf{s} = s_1 s_2 \dots s_n$  with  $s_1 = 0$  and  $0 \leq s_{i+1} \leq \max\{s_j\}_{j=1}^i + 1$ , for all  $i$ ,  $1 \leq i \leq n-1$ . We denote by  $R_n$  the set of length  $n$  restricted growth functions, and its cardinality is given by the  $n$ th Bell number (sequence A000110 in [6]), with the exponential generating function  $e^{e^x} - 1$ . And length  $n$  restricted growth functions encode the partitions of an  $n$ -set.

For an integer  $b \geq 1$ , let  $R_n(b)$  denote the set of  $b$ -bounded sequences in  $R_n$ , that is,

$$R_n(b) = \{s_1 s_2 \dots s_n \in R_n : \max\{s_i\}_{i=1}^n \leq b\},$$

and

$$R_n^*(b) = \{s_1 s_2 \dots s_n \in R_n : \max\{s_i\}_{i=1}^n = b\}.$$

See Table 1 for an example.

1.	0 0 0 0 0	15.	0 1 0 0 0	<i>3</i>	29.	<b>0 1 1 1 2</b>	<i>1</i>	
2.	0 0 0 0 1	<i>1</i>	16.	0 1 0 0 1	<i>1</i>	30.	<b>0 1 1 2 2</b>	<i>1</i>
3.	0 0 0 1 0	<i>2</i>	17.	<b>0 1 0 0 2</b>	<i>1</i>	31.	<b>0 1 1 2 1</b>	<i>1</i>
4.	0 0 0 1 1	<i>1</i>	18.	0 1 0 1 0	<i>2</i>	32.	<b>0 1 1 2 0</b>	<i>1</i>
5.	<b>0 0 0 1 2</b>	<i>1</i>	19.	0 1 0 1 1	<i>1</i>	33.	<b>0 1 2 2 0</b>	<i>1</i>
6.	0 0 1 0 0	<i>3</i>	20.	<b>0 1 0 1 2</b>	<i>1</i>	34.	<b>0 1 2 2 1</b>	<i>1</i>
7.	0 0 1 0 1	<i>1</i>	21.	<b>0 1 0 2 2</b>	<i>1</i>	35.	<b>0 1 2 2 2</b>	<i>1</i>
8.	<b>0 0 1 0 2</b>	<i>1</i>	22.	<b>0 1 0 2 1</b>	<i>1</i>	36.	<b>0 1 2 1 2</b>	<i>1</i>
9.	0 0 1 1 0	<i>2</i>	23.	<b>0 1 0 2 0</b>	<i>1</i>	37.	<b>0 1 2 1 1</b>	<i>1</i>
10.	0 0 1 1 1	<i>1</i>	24.	0 1 1 0 0	<i>2</i>	38.	<b>0 1 2 1 0</b>	<i>1</i>
11.	<b>0 0 1 1 2</b>	<i>1</i>	25.	0 1 1 0 1	<i>1</i>	39.	<b>0 1 2 0 2</b>	<i>2</i>
12.	<b>0 0 1 2 2</b>	<i>1</i>	26.	<b>0 1 1 0 2</b>	<i>1</i>	40.	<b>0 1 2 0 1</b>	<i>1</i>
13.	<b>0 0 1 2 1</b>	<i>1</i>	27.	0 1 1 1 0	<i>2</i>	41.	<b>0 1 2 0 0</b>	<i>1</i>
14.	<b>0 0 1 2 0</b>	<i>1</i>	28.	0 1 1 1 1	<i>1</i>			

Table 1: The set  $R_5(2)$ , and in bold-face the set  $R_5^*(2)$ . Sequences are listed in  $\prec$  order (see Definition 2) and in italic is the Hamming distance between consecutive sequences.

If a list of same length sequences is such that the Hamming distance between successive sequences (that is, the number of positions in which the sequences differ) is bounded from above by a constant, independent on the sequences length, then the list is said to be a *Gray code*. When we want to explicitly specify this constant, say  $d$ , then we refer to such a list as a  $d$ -*Gray code*; in addition, if the positions where the successive sequences differ are adjacent, then we say that the list is a  $d$ -*adjacent Gray code*.

The next two definitions give order relations on the set of  $m$ -ary integer sequences of length  $n$  on which our Gray codes are based.

**Definition 1.** Let  $m$  and  $n$  be positive integers with  $m \geq 2$ . The *Reflected Gray Code Order*  $\prec$  on  $\{0, 1, \dots, m-1\}^n$  is defined as:  $\mathbf{s} = s_1 s_2 \dots s_n$  is less than  $\mathbf{t} = t_1 t_2 \dots t_n$ , denoted by  $\mathbf{s} \prec \mathbf{t}$ , if

$$\text{either } \sum_{i=1}^{k-1} s_i \text{ is even and } s_k < t_k, \text{ or } \sum_{i=1}^{k-1} s_i \text{ is odd and } s_k > t_k$$

for some  $k$  with  $s_i = t_i$  ( $1 \leq i \leq k-1$ ) and  $s_k \neq t_k$ .

This order relation is the natural extension to  $m$ -ary sequences of the order induced by the Binary Reflected Gray Code introduced in [2]. See for example [1, 4] where this order relation and its variations are considered in the context of factor avoiding words and of statistic-restricted sequences.

**Definition 2.** Let  $m$  and  $n$  be positive integers with  $m \geq 2$ . The *co-Reflected Gray Code Order*<sup>1</sup>  $\prec$  on  $\{0, 1, \dots, m-1\}^n$  is defined as:  $\mathbf{s} = s_1 s_2 \dots s_n$  is less than  $\mathbf{t} = t_1 t_2 \dots t_n$ , denoted by  $\mathbf{s} \prec \mathbf{t}$ , if

$$\text{either } U_k \text{ is even and } s_k < t_k, \text{ or } U_k \text{ is odd and } s_k > t_k$$

for some  $k$  with  $s_i = t_i$  ( $1 \leq i \leq k-1$ ) and  $s_k \neq t_k$  where  $U_k = |\{i \in \{1, 2, \dots, k-1\} : s_i \neq 0, s_i \text{ is even}\}|$ .

See Table 2 for an example.

For a set  $S$  of same length integer sequences the  $\prec$ -*first* (resp.  $\prec$ -*last*) sequence in  $S$  is the first (resp. last) sequence when the set is listed in  $\prec$  order; and  $\prec$ -*first* and  $\prec$ -*last* are defined in a similar way. And for sequence  $\mathbf{u}$ ,  $\mathbf{u} | S$  denotes the subset of  $S$  of sequences having prefix  $\mathbf{u}$ .

Both order relations, Reflected and co-Reflected Gray Code Order produce prefix partitioned lists, that is to say, if a set of sequences is listed in one of these order relations, then the sequences having a common prefix are consecutive in the list.

1.	0 0 0	10.	1 0 0	19.	2 2 0
2.	0 0 1	11.	1 0 1	20.	2 2 1
3.	0 0 2	12.	1 0 2	21.	2 2 2
4.	0 1 0	13.	1 1 0	22.	2 1 2
5.	0 1 1	14.	1 1 1	23.	2 1 1
6.	0 1 2	15.	1 1 2	24.	2 1 0
7.	0 2 2	16.	1 2 2	25.	2 0 2
8.	0 2 1	17.	1 2 1	26.	2 0 1
9.	0 2 0	18.	1 2 0	27.	2 0 0

Table 2: The set  $\{0, 1, 2\}^3$  listed in  $\prec$  order.

### 3 The Gray codes

In this section we show that the set  $R_n(b)$ , with  $b$  odd, listed in  $\prec$  order is a Gray code. However,  $\prec$  does not induce a Gray code when  $b$  is even: the Hamming distance between two consecutive sequences can be arbitrary large for large enough  $n$ . To overcome this, we consider  $\prec$  order instead of  $\prec$  order when  $b$  is even, and we show that the obtained list is a Gray code.

In the proof of Theorem 1 below we need the following propositions which give the forms of the last and first sequence in  $R_n(b)$  having a certain fixed prefix, when sequences are listed in  $\prec$  order.

**Proposition 1.** *Let  $b \geq 1$  and odd,  $k \leq n-2$  and  $\mathbf{s} = s_1 \dots s_k$ . If  $\mathbf{t}$  is the  $\prec$ -last sequence in  $\mathbf{s} | R_n(b)$ , then  $\mathbf{t}$  has one of the following forms:*

<sup>1</sup>In [4] a similar terminology is used for a slightly different notion

1.  $\mathbf{t} = \mathbf{s}M0\dots 0$  if  $\sum_{i=1}^k s_i$  is even and  $M$  is odd,
2.  $\mathbf{t} = \mathbf{s}M(M+1)0\dots 0$  if  $\sum_{i=1}^k s_i$  is even and  $M$  is even,
3.  $\mathbf{t} = \mathbf{s}0\dots 0$  if  $\sum_{i=1}^k s_i$  is odd,

where  $M = \min\{b, \max\{s_i\}_{i=1}^k + 1\}$ .

*Proof.* Let  $\mathbf{t} = s_1 \dots s_k t_{k+1} \dots t_n$  be the  $\prec$ -last sequence in  $\mathbf{s} \mid R_n(b)$ .

Referring to the definition of  $\prec$  order in Definition 1, if  $\sum_{i=1}^k s_i$  is even, then  $t_{k+1} = \min\{b, \max\{s_i\}_{i=1}^k + 1\} = M$ , and based on the parity of  $M$ , two cases can occur.

- If  $M$  is odd, then we have that  $\sum_{i=1}^k s_i + t_{k+1} = \sum_{i=1}^k s_i + M$  is odd, thus  $t_{k+2} \dots t_n = 0 \dots 0$ , and we retrieve the form prescribed by the first point of the proposition.
- If  $M$  is even, then  $M \neq b$  and  $\sum_{i=1}^k s_i + t_{k+1} = \sum_{i=1}^k s_i + M$  is even, thus  $t_{k+2} = \max\{s_1, \dots, s_k, t_{k+1}\} + 1 = M + 1$ , which is odd. Next, we have  $\sum_{i=1}^k s_i + t_{k+1} + t_{k+2} = \sum_{i=1}^k s_i + 2M + 1$  is odd, and this implies as above that  $t_{k+3} \dots t_n = 0 \dots 0$ , and we retrieve the second point of the proposition.

For the case when  $\sum_{i=1}^k s_i$  is odd, in a similar way we have  $t_{k+1} \dots t_n = 0 \dots 0$ . □

The next proposition is the ‘first’ counterpart of the previous one. Its proof is similar by exchanging the parity of the summation from ‘odd’ to ‘even’ and vice-versa, and it is left to the reader.

**Proposition 2.** *Let  $b \geq 1$  and odd,  $k \leq n - 2$  and  $\mathbf{s} = s_1 \dots s_k$ . If  $\mathbf{t}$  is the  $\prec$ -first sequence in  $\mathbf{s} \mid R_n(b)$ , then  $\mathbf{t}$  has one of the following forms:*

1.  $\mathbf{t} = \mathbf{s}M0\dots 0$  if  $\sum_{i=1}^k s_i$  is odd and  $M$  is odd,
2.  $\mathbf{t} = \mathbf{s}M(M+1)0\dots 0$  if  $\sum_{i=1}^k s_i$  is odd and  $M$  is even,
3.  $\mathbf{t} = \mathbf{s}0\dots 0$  if  $\sum_{i=1}^k s_i$  is even,

where  $M = \min\{b, \max\{s_i\}_{i=1}^k + 1\}$ .

Based on Propositions 1 and 2, we have the following theorem.

**Theorem 1.** *For any  $n, b \geq 1$  and  $b$  odd,  $R_n(b)$  listed in  $\prec$  order is a 3-adjacent Gray code.*

*Proof.* Let  $\mathbf{s} = s_1 s_2 \dots s_n$  and  $\mathbf{t} = t_1 t_2 \dots t_n$  be two consecutive sequences in  $\prec$  ordered list for the set  $R_n(b)$ , with  $\mathbf{s} \prec \mathbf{t}$ , and let  $k$  be the leftmost position where  $\mathbf{s}$  and  $\mathbf{t}$  differ. If  $k \geq n - 2$ , then obviously  $\mathbf{s}$  and  $\mathbf{t}$  differ in at most three positions, otherwise let  $\mathbf{s}' = s_1 \dots s_k$  and  $\mathbf{t}' = t_1 \dots t_k$ . Thus,  $\mathbf{s}$  is the  $\prec$ -last sequence in  $\mathbf{s}' \mid R_n(b)$  and  $\mathbf{t}$  is the  $\prec$ -first sequence in  $\mathbf{t}' \mid R_n(b)$ . Combining Propositions 1 and 2 we have that, when  $k \leq n - 3$ ,  $s_{k+3} s_{k+4} \dots s_n = t_{k+3} t_{k+4} \dots t_n = 00 \dots 0$ . And since  $s_i = t_i$  for  $i = 1 \dots, k - 1$ , the statement holds. □

Theorem 3 below shows the Graycodeness of  $R_n(b)$ ,  $b \geq 1$  and even, listed in  $\prec$  order, and as for Theorem 1 we need the next two propositions; in its proof we will make use of the Iverson bracket notation:  $[P]$  is 1 if the statement  $P$  is true, and 0 otherwise. Thus, for a sequence  $s_1 s_2 \dots s_n$  and a  $k \leq n$ ,  $|\{i \in \{1, 2, \dots, k\} : s_i \neq 0 \text{ and } s_i \text{ is even}\}| = \sum_{i=1}^k [s_i \neq 0 \text{ and } s_i \text{ is even}]$ .

**Proposition 3.** Let  $b \geq 2$  and even,  $k \leq n - 2$  and  $\mathbf{s} = s_1 s_2 \dots s_k$ . If  $\mathbf{t}$  is the  $\prec$ -last sequence in  $\mathbf{s} \mid R_n(b)$ , then  $\mathbf{t}$  has one of the following forms:

1.  $\mathbf{t} = \mathbf{s}M0\dots 0$  if  $U_{k+1}$  is even and  $M$  is even,
2.  $\mathbf{t} = \mathbf{s}M(M+1)0\dots 0$  if  $U_{k+1}$  is even and  $M$  is odd,
3.  $\mathbf{t} = \mathbf{s}0\dots 0$  if  $U_{k+1}$  is odd,

where  $M = \min\{b, \max\{s_i\}_{i=1}^k + 1\}$  and  $U_{k+1} = \sum_{i=1}^k [s_i \neq 0 \text{ and } s_i \text{ is even}]$ .

*Proof.* Let  $\mathbf{t} = s_1 \dots s_k t_{k+1} \dots t_n$  be the  $\prec$ -last sequence in  $\mathbf{s} \mid R_n(b)$ .

Referring to the definition of  $\prec$  order in Definition 2, if  $U_{k+1}$  is even, then  $t_{k+1} = \min\{b, \max\{s_i\}_{i=1}^k + 1\} = M > 0$ , and based on the parity of  $M$ , two cases can occur.

- If  $M$  is even, then  $U_{k+1} + [t_{k+1} \neq 0 \text{ and } t_{k+1} \text{ is even}] = U_{k+1} + 1$  is odd, thus  $t_{k+2} \dots t_n = 0 \dots 0$ , and we retrieve the form prescribed by the first point of the proposition.
- If  $M$  is odd, then  $M \neq b$  and  $U_{k+1} + [t_{k+1} \neq 0 \text{ and } t_{k+1} \text{ is even}] = U_{k+1}$  is even, thus  $t_{k+2} = \max\{s_1, \dots, s_k, t_{k+1}\} + 1 = M + 1$ , which is even. Next, we have  $U_{k+1} + [t_{k+1} \neq 0 \text{ and } t_{k+1} \text{ is even}] + [t_{k+2} \neq 0 \text{ and } t_{k+2} \text{ is even}] = U_{k+1} + 1$  is odd, and this implies as above that  $t_{k+3} \dots t_n = 0 \dots 0$ , and we retrieve the second point of the proposition.

For the case when  $U_{k+1}$  is odd, in a similar way we have  $t_{k+1} \dots t_n = 0 \dots 0$ . □

The next proposition is the ‘first’ counterpart of the previous one.

**Proposition 4.** Let  $b \geq 2$  and even,  $k \leq n - 2$  and  $\mathbf{s} = s_1 s_2 \dots s_k$ . If  $\mathbf{t}$  is the  $\prec$ -first sequence in  $\mathbf{s} \mid R_n(b)$ , then  $\mathbf{t}$  has one of the following forms:

1.  $\mathbf{t} = \mathbf{s}M0\dots 0$  if  $U_{k+1}$  is odd and  $M$  is even,
2.  $\mathbf{t} = \mathbf{s}M(M+1)0\dots 0$  if  $U_{k+1}$  is odd and  $M$  is odd,
3.  $\mathbf{t} = \mathbf{s}0\dots 0$  if  $U_{k+1}$  is even,

where  $M = \min\{b, \max\{s_i\}_{i=1}^k + 1\}$  and  $U_{k+1} = \sum_{i=1}^k [s_i \neq 0 \text{ and } s_i \text{ is even}]$ .

Based on Propositions 3 and 4 we have the following theorem, its proof is similar with that of Theorem 1.

**Theorem 2.** For any  $n \geq 1$ ,  $b \geq 2$  and even,  $R_n(b)$  listed in  $\prec$  order is a 3-adjacent Gray code.

It is worth to mention that, neither  $\prec$  for even  $b$ , nor  $\prec$  for odd  $b$  yields a Gray code on  $R_n(b)$ . Considering  $b \geq n$  in Theorem 1 and 2, the bound  $b$  does not actually provide any restriction, and in this case  $R_n(b) = R_n$ , and we have the following corollary.

**Corollary 1.** For any  $n \geq 1$ ,  $R_n$  listed in both  $\prec$  and  $\prec$  order are 3-adjacent Gray codes.

**Theorem 3.** For any  $b \geq 1$  and odd,  $n > b$ ,  $R_n^*(b)$  listed in  $\prec$  order is a 5-Gray code.

*Proof.* For two integers  $a$  and  $b$ ,  $0 < a \leq b$ , we define  $\tau_{a,b}$  as the length  $b - a$  increasing sequence  $(a + 1)(a + 2) \dots (b - 1)b$ , and  $\tau_{a,b}$  is vanishingly empty if  $a = b$ . Imposing to a sequence  $\mathbf{s}$  in  $R_n(b)$  to have its largest element equal to  $b$  (so, to belong to  $R_n^*(b)$ ) implies that either  $b$  occurs in  $\mathbf{s}$  before its last position, or  $\mathbf{s}$  ends with  $b$ , and in this case the tail of  $\mathbf{s}$  is  $\tau_{a,b}$  for an appropriate  $a < b$ . More precisely, in the latter case,  $\mathbf{s}$  has the form  $s_1 s_2 \dots s_j \tau_{a,b}$ , for some  $j$  and  $a$ , with  $a = \max\{s_i\}_{i=1}^j$  and  $j = n - (b - a)$ .

Now let  $\mathbf{s} = s_1 s_2 \dots s_n \prec \mathbf{t} = t_1 t_2 \dots t_n$  be two consecutive sequences in the  $\prec$  ordered list for  $R_n^*(b)$ , and let  $k \leq n - 3$  be the leftmost position where  $\mathbf{s}$  and  $\mathbf{t}$  differ, thus  $s_1 s_2 \dots s_{k-1} = t_1 t_2 \dots t_{k-1}$ . It follows that  $\mathbf{s}$  is the  $\prec$ -last sequence in  $R_n^*(b)$  having the prefix  $s_1 s_2 \dots s_k$ , and using Proposition 1 and the notations therein, by imposing that  $\max\{s_i\}_{i=1}^n$  is equal to  $b$ , we have:

- if  $\sum_{i=1}^k s_i$  is odd, then  $\mathbf{s}$  has the form  $s_1 s_2 \dots s_k 0 \dots 0 \tau_{a,b}$ , where  $a = \max\{s_i\}_{i=1}^k$ ,
- if  $\sum_{i=1}^k s_i$  is even, then  $\mathbf{s}$  has one of the following forms:
  - $s_1 s_2 \dots s_k M 0 \dots 0 \tau_{M,b}$ , or
  - $s_1 s_2 \dots s_k M(M + 1) 0 \dots 0 \tau_{M+1,b}$ .

When the above  $\tau$ 's suffixes are empty, we retrieve precisely the three cases in Proposition 1.

Similarly,  $\mathbf{t}$  is the  $\prec$ -first sequence in  $R_n^*(b)$  having the prefix  $t_1 t_2 \dots t_{k-1} t_k = s_1 s_2 \dots s_{k-1} t_k$ . Since by the definition of  $\prec$  order we have that  $t_k = s_k + 1$  or  $t_k = s_k - 1$ , it follows that  $\sum_{i=1}^k t_i$  and  $\sum_{i=1}^k s_i$  have different parity (that is,  $\sum_{i=1}^k t_i$  is odd if and only if  $\sum_{i=1}^k s_i$  is even), and by Proposition 2 and replacing for notational convenience  $M$  by  $M'$ , we have:

- if  $\sum_{i=1}^k s_i$  is odd, then  $\mathbf{t}$  has the form  $t_1 t_2 \dots t_k 0 \dots 0 \tau_{a',b}$ , where  $a' = \max\{t_i\}_{i=1}^k$ ,
- if  $\sum_{i=1}^k s_i$  is even, then  $\mathbf{t}$  has one of the following forms:
  - $t_1 t_2 \dots t_k M' 0 \dots 0 \tau_{M',b}$ , or
  - $t_1 t_2 \dots t_k M'(M' + 1) 0 \dots 0 \tau_{M'+1,b}$ .

With these notations, since  $t_k \in \{s_k + 1, s_k - 1\}$ , it follows that

- if  $\sum_{i=1}^k s_i$  is odd, then  $a' \in \{a - 1, a, a + 1\}$ , and so the length of  $\tau_{a,b}$  and that of  $\tau_{a',b}$  differ by at most one; and
- if  $\sum_{i=1}^k s_i$  is even, then  $M' \in \{M - 1, M, M + 1\}$ , and the length of the non-zero tail of  $\mathbf{s}$  and that of  $\mathbf{t}$  (defined by means of  $\tau$  sequences) differ by at most two.

Finally, the whole sequences  $\mathbf{s}$  and  $\mathbf{t}$  differ in at most five (not necessarily adjacent) positions, and the statement holds.  $\square$

## 4 Generating algorithms

An exhaustive generating algorithm is one generating all sequences in a combinatorial class, with some predefined properties (*e.g.*, having the same length). Such an algorithm is said to run in *constant amortized time* if it generates each object in  $O(1)$  time, in amortized sense. In [3] the author called such an algorithm *CAT algorithm* and shows that a recursive generating algorithm satisfying the following properties is precisely a CAT algorithm:

- Each recursive call either generates an object or produces at least two recursive calls;
- The amount of computation in each recursive call is proportional to the degree of the call (that is, to the number of subsequent recursive calls produced by current call).

Procedure GEN1 in Fig. 1 generates all sequences belonging to  $R_n(b)$  in Reflected Gray Code Order. Especially when  $b$  is odd, the generation induces a 3-adjacent Gray code. The bound  $b$  and the generated sequence  $s = s_1 s_2 \dots s_n$  are global. The  $k$  parameter is the position where the value is to be assigned (see line 8 and 13); the  $dir$  parameter represents the direction of sequencing for  $s_k$ , whether it is up (when  $dir$  is even, see line 7) or down (when  $dir$  is odd, see line 12); and  $m$  is such that  $m + 1$  is the the maximum value that can be assigned to  $s_k$ , that is,  $\min\{b - 1, \max\{s_i\}_{i=1}^{k-1}\}$  (see line 5).

The algorithm initially sets  $s_1 = 0$ , and the recursive calls are triggered by the initial call GEN1(2,0,0). For the current position  $k$ , the algorithm assigns a value to  $s_k$  (line 8 or 13) followed by recursive calls in line 10 or 15. This scheme guarantees that each recursive call will produce subsequent recursive calls until  $k = n + 1$  (line 4), that is, when a sequence of length  $n$  is generated and printed out by TYPE() procedure. This process eventually generates all sequences in  $R_n(b)$ . In addition, by construction, algorithm GEN1 satisfies the previous CAT desiderata, and so it is an efficient exhaustive generating algorithm.

```

01 procedure GEN1( $k, dir, m$ : integer)
02 global  $s, n, b$ : integer;
03 local  $i, u$ : integer;
04 if  $k = n + 1$  then TYPE();
05 else if  $m = b$  then  $m := b - 1$ ; endif
06     if  $dir \bmod 2 = 0$ 
07     then for  $i := 0$  to  $m + 1$  do
08          $s_k := i$ ;
09         if  $m < s_k$  then  $u := s_k$ ; else  $u := m$ ; endif
10         GEN1( $k + 1, i, u$ );
11     endfor
12     else for  $i := m + 1$  downto 0 do
13          $s_k := i$ ;
14         if  $m < s_k$  then  $u := s_k$ ; else  $u := m$ ; endif
15         GEN1( $k + 1, i + 1, u$ );
16     endfor
17     endif
18 endif
19 end procedure.

```

Figure 1: Reflected Gray Code Order generating algorithm for  $R_n(b)$ ; it produces a 3-Gray code when  $b$  is odd.

Similarly, the call GEN2(2,0,0) of the algorithm in Fig. 2 generates sequences in  $R_n(b)$  in co-Reflected Gray Code Order, and in particular when  $b$  is even, a 3-adjacent Gray code for these sequences. And again it satisfies the CAT desiderata, and so it is an efficient exhaustive generating algorithm.

Finally, algorithm GEN3 in Fig. 3 generates the set  $R_n^*(b)$  in Reflected Gray Code Order and produces a 5-Gray code if  $b$  is odd. It mimics algorithm GEN1 and the only differences consist in an additional parameter  $a$  and lines 5, 6, 13 and 19, and its main call is GEN3(2, 0, 0, 0). Parameter  $a$  keeps track of the maximum value in the prefix  $s_1 s_2 \dots s_{k-1}$  of the currently generated sequence, and it is updated in lines 13 and 19. Furthermore, when the current position  $k$  belongs to a  $\tau$ -tail (see the proof of Theorem 3), that is, condition  $k = n + 1 + a - b$  in line 5 is satisfied, then the imposed value is written in this position, and similarly for the next two positions. Theorem 3 ensures that there are no differences between the current sequence and the previous generated one beyond position  $k + 2$ , and thus a new sequence in  $R_n^*(b)$  is generated. And as previously, GEN3 is a CAT generating algorithm.

```

procedure GEN2( $k, dir, m$ : integer)
global  $s, n, b$ : integer;
local  $i, u$ : integer;
if  $k = n + 1$  then TYPE();
else if  $m = b$  then  $m := b - 1$ ; endif
  if  $dir \bmod 2 = 0$ 
    then for  $i := 0$  to  $m + 1$  do
       $s_k := i$ ;
      if  $m < s_k$  then  $u := s_k$ ; else  $u := m$ ; endif
      if  $s_k = 0$  then GEN2( $k + 1, 0, u$ );
        else GEN2( $k + 1, i + 1, u$ );
      endif
    endfor
  else for  $i := m + 1$  downto 0 do
     $s_k := i$ ;
    if  $m < s_k$  then  $u := s_k$ ; else  $u := m$ ; endif
    if  $s_k = 0$  then GEN2( $k + 1, 1, u$ );
      else GEN2( $k + 1, i, u$ );
    endif
  endfor
endif
end procedure.

```

Figure 2: Co-Reflected Gray Code Order generating algorithm for  $R_n(b)$ ; it produces a 3-Gray code when  $b$  is even.

**Final remarks.** We suspect that the upper bounds 3 in Theorems 1 and 2, and 5 in Theorem 3 are not tight, and a natural question arises: are there more restrictive Gray codes for  $R_n(b)$  and for  $R_n^*(b)$  with  $b$  odd? Finally, is there a natural order relation inducing a Gray code on  $R_n^*(b)$  when  $b$  is even?



```

01 procedure GEN3(k, dir, m, a: integer)
02 global s, n, b: integer;
03 local i, u, l: integer;
04 if k = n + 1 then TYPE();
05 else if k = n + 1 + a - b
06     then for i := 0 to 2 do if k + i ≤ n then sk+i := a + 1 + i; endif endfor
07         TYPE();
08     else if m = b then m := b - 1; endif
09         if dir mod 2 = 0
10             then for i := 0 to m + 1 do
11                 sk := i;
12                 if m < sk then u := sk; else u := m; endif
13                 if a < sk then l := sk; else l := a; endif
14                 GEN3(k + 1, i, u, l);
15             endfor
16         else for i := m + 1 downto 0 do
17             sk := i;
18             if m < sk then u := sk; else u := m; endif
19             if a < sk then l := sk; else l := a; endif
20             GEN3(k + 1, i + 1, u, l);
21         endfor
22     endif
23 endif
24 endif
25 end procedure.

```

Figure 3: Generating algorithm for  $R_n^*(b)$ ,  $n > b \geq 1$ , with respect to Reflected Gray Code Order; it produces a 5-Gray code when  $b$  is odd.

## References

- [1] A. Bernini, S. Bilotta, R. Pinzani, A. Sabri, V. Vajnovszki, Reflected Gray codes for  $q$ -ary words avoiding a given factor, *Acta Informatica*, 52(7), 573-592 (2015).
- [2] F. Gray, Pulse code communication, U.S. Patent 2632058 (1953).
- [3] F. Ruskey, Combinatorial generation, Book in preparation.
- [4] A. Sabri, V. Vajnovszki, Reflected Gray code based orders on some restricted growth sequences, *The Computer Journal*, 58(5), 1099-1111 (2015).
- [5] A. Sabri, V. Vajnovszki, Bounded growth functions: Gray codes and exhaustive generation, *The Japanese Conference on Combinatorics and its Applications*, May 21-25, 2016, Kyoto, Japan.
- [6] N.J.A. Sloane, The On-line Encyclopedia of Integer Sequences, available electronically at <http://oeis.org>.