PU. M. A. Vol. **21** (2010), No. 1, pp. 27 – 38

A Gray code for permutations of size $2d$ with d descents

ELISABETTA GRAZZINI
Dipartimento di Sistemi e Informatica
Università degli Studi di Firenze
Firenze, Italy
e-mail: elisabetta.grazzini@unifi.it

and
ELISA PERGOLA
Dipartimento di Sistemi e Informatica
Università degli Studi di Firenze
Firenze, Italy
e-mail: elisa@dsi.unifi.it

(Received: January 1, 2010, and in revised form June 1, 2010)

Abstract. Our aim is to define a Gray code for the class of minimal permutations with d descents and size $2d$ enumerated by Catalan numbers. We consider two algorithms for generating a Gray code for Catalan structures, both of them based on the common strategy of reflecting subtrees in the computation tree. The application of these algorithms to minimal permutations of size $2d$ with d descents does not maintain the distance between two consecutive permutations constant. In this paper we propose a new version of these algorithms and prove that two consecutive permutations differ in at most three positions, independently of their size.

Keywords: Gray codes, permutations, exhaustive generation, tandem duplication - random loss model, reflectable languages, combinatorial problems.

1 Introduction

A permutation of size n is a bijective map from $[1..n]$ to itself. We denote by S_n the set of permutations of size n. We consider a permutation $\sigma \in S_n$ as the word $\sigma_1\sigma_2\dots\sigma_n$ of n letters on the alphabet $\{1, 2, \dots, n\}$, containing each letter exactly once (we often use the word *element* instead of letter). For example, 624351 represents the permutation $\sigma \in S_6$ such that $\sigma_1 = 6, \sigma_2 = 2, \dots, \sigma_6 = 1$.

An *increasing substring* of σ is just a sequence of consecutive elements of σ that are in increasing order. An increasing substring is maximal if it can be extended neither on the left nor on the right.

In [6], instead of the number of maximal increasing substrings, it was introduced the number of descents which is a very well-known statistics on permutations.

DEFINITION 1.1 Given a permutation σ of size n, we say that there is a *descent* (resp. *ascent*) in the position i, $1 \leq i \leq n-1$, if $\sigma_i > \sigma_{i+1}$ (resp. $\sigma_i < \sigma_{i+1}$). A descent (resp.ascent) is *isolated* if it is preceded and followed by an ascent (resp. descent).

EXAMPLE 1.2 For example, $\sigma = 698413725$ has 4 descents, namely at positions 2, 3, 4, 7.

DEFINITION 1.3 A permutation $\pi \in S_k$ is a *pattern* of a permutation $\sigma \in S_n$ if there is a subsequence of σ which is order-isomorphic to π; i.e., if there is a subsequence $\sigma_{i_1}\sigma_{i_2}\dots\sigma_{i_k}$ of σ (with $1 \leq i_1 < i_2 < \dots < i_k \leq n$) such that $\sigma_{i_\ell} < \sigma_{i_m}$ whenever $\pi_\ell < \pi_m$.
We also say that π is *involved* in σ and call $\sigma_{i_1}\sigma_{i_2}\dots\sigma_{i_k}$ an *occurrence* of π in σ.

We write $\pi \prec \sigma$ to denote that π is a pattern of σ. A permutation σ that does not contain π as a pattern is said to *avoid* π. The class of all permutations avoiding the patterns $\pi_1, \pi_2 \dots \pi_k$ is denoted $S(\pi_1, \pi_2, \dots, \pi_k)$. We say that $S(\pi_1, \pi_2, \dots, \pi_k)$ is a class of pattern-avoiding permutations of *basis* $\{\pi_1, \pi_2, \dots, \pi_k\}$.

1.1 Minimal permutations with d descents

In the *whole genome duplication - random loss model* genomes composed of n genes are modelled by permutations on $[1..n]$, that can evolve through *duplication-loss steps*, [5]. Each of these steps is composed of two elementary operations. Firstly, the permutation (a fragment of consecutive elements of it for the tandem duplication) is duplicated, and the duplicated copy is inserted immediately after the original copy: this is the *whole genome duplication*. Then the *random loss* occurs: one copy of every duplicated element is lost, so that we get a permutation at the end of the step. In [6] it was shown that the class of permutations obtained in this model after a given number p of steps is a class of pattern-avoiding permutations of finite basis; in particular it was proved the following theorem.

THEOREM 1.4 *The class of permutations obtainable in at most p steps in the whole genome duplication - random loss model is a class of pattern-avoiding permutations whose basis $\mathcal{B}_d$ is finite and is composed of the minimal permutations with $d = 2^p$ descents, minimal being intended in the sense of $\prec$.*

In this paper, we focus on the basis $\mathcal{B}_d$ of excluded patterns appearing in Theorem 1.4. More generally, we do not assume that d is a power of 2. From here on, by minimal permutation with d descents, we mean a permutation that is minimal in the sense of the pattern-involvement relation $\prec$ for the property of having d descents.

EXAMPLE 1.5 Let $\sigma = 7\,4\,1\,3\,2\,5\,8\,6\,9$ be a permutation with 4 descents; σ is not minimal with 4 descents. Indeed, the elements 1 and 5 can be removed from σ without changing the number of descents. Doing this, we obtain permutation $\pi = 5\,3\,2\,1\,6\,4\,7$ which is minimal with 4 descents: it is impossible to remove an element from it while preserving the number of descents equal to 4. However, π is not of minimal *size* among the permutation with 4 descents: π has size 7 whereas permutation $5\,4\,3\,1\,2$ has 4 descents but size 5.

A more exhaustive characterization of minimal permutations is given in [5] and it can be summarized in the following theorem giving a local characterization of minimal permutations with d descents.

THEOREM 1.6 *A permutation σ of size n is minimal with d descents if and only if it has exactly d descents and its ascents $\sigma_i\sigma_{i+1}$ are such that $2 \leq i \leq n-2$ and $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ forms an occurrence of either the pattern 2143 or the pattern 3142.*

Here we are interested only in minimal permutations with d descents and size $2d$. Indeed, they can have neither two consecutive ascents, nor two consecutive descents, otherwise it would be impossible to reach size $2d$. Consequently, they all result from of an alternation of isolated descents and isolated ascents, of course starting and ending with a descent. Therefore, such a permutation always has 1 as its second element and $2d$ in position $2d-1$.

EXAMPLE 1.7 The permutation $\sigma = 41627385$, of size 8, has 4 descents, namely 4, 6, 7, 8.

In the sequel we will only refer to minimal permutations of size $2d$ with d descents, simply named "$2d$ minimal permutations".

1.2 Outline of the paper

In this paper, we take into consideration two algorithms, developed in [4] and [8], for generating $2d$ minimal permutations in Gray code order. We first prove that by using the original versions of the two algorithms, the distance between two consecutive permutations is not constant. Then we propose a new version of these algorithms and prove that two consecutive permutations differ in at most three positions, independently of their size.

In the next section we recall the ECO construction for these permutations firstly established in [5], proving that these permutations are enumerated by Catalan numbers. The generating algorithms and their improved version are described in Sections 3 and 4, where we also prove that two consecutive permutations in the list differ in at most three positions.

2 ECO generation of $2d$ minimal permutations

In this section we describe an ECO contruction for $2d$ minimal permutations that (though different) turns out to be the same as to the one provided in [5]. The ECO method [2] obtains combinatorial objects of size n from those of size $n-1$, through a process of *local expansion*, whereby the objects are modified only by the addition of an elementary block of objects. Then, once the way of expanding the objects ($2d$ minimal permutations in this case) is defined, a *label* is given to each object. The label of an object is the number of its *sons*, that is the number of objects that are obtained from it through the local expansion process. Those sons can again receive a label by the same method. The infinite tree in which any permutation is the father of its children is called the *generating tree* of the combinatorial class [1, 7].

By applying the ECO labelling method of the combinatorial objects, it is often possible to derive a *succession rule*. This rule describes the production (in terms of labels) of the possible labels in the generating tree, together with a starting point. The usual notation for a succession rule is the following:

$$\Omega : \begin{cases} (a) \\ (k) \rightsquigarrow (e_1(k))(e_2(k))\dots(e_k(k)) \qquad k \in \mathbb{N}\ . \end{cases}$$

The axiom (a) is the root of the generating tree corresponding to the minimal size combinatorial object and each node with label (k) generates k sons with labels $(e_1(k)), (e_2(k)), \dots, (e_k(k))$ (each $e_i(k)$ is an integer) [1, 7].

The ECO label that is given to a $2d$ minimal permutation σ for this purpose is $(2d - \sigma_{2d} + 1)$, σ_{2d} being the element of σ in position $2d$. The growth parameter for $2d$ minimal permutations is the number d of descents.

Consider a $2d$ minimal permutation σ with d descents and ECO label (k). Its sons are the minimal permutations (of size $2d + 2$) with $d + 1$ descents obtained by adding to the right of σ the element $2d+2$, in position $2d+1$, and i in position$2d$, for $2d+2-k \leq i \leq 2d+1$. The elements j in σ with $j \geq i$ are turned into $j+1$ in order to maintain both the relative order of the elements of σ and the property that all the integers of $[1..2d+2]$ are present in the new permutations exactly once.

Since $k = 2d - \sigma_{2d} + 1$, it is easy to check that all the $2d$ minimal permutations obtained in this way are correct, and that all of them are obtained.

Example 2.1 The minimal permutation with 2 descents 2143 has ECO label (2) and its sons are the two minimal permutations with 3 descents 214365 and 215364.

The ECO labels of the k sons of a $2d$ minimal permutation with d descents and with ECO label (k) are, by the above formula, $(2(d+1) - i + 1)$ with $2d + 2 - k \leq i \leq 2d + 1$, that is to say the sons have labels $(2), (3), \ldots (k), (k+1)$.

The starting point for this ECO construction is the minimal permutation 21 whose ECO label is (2).

Then, the succession rule obtained for this ECO construction of minimal permutation is

$$\left\{ \begin{array}{l} (2) \\ (k) \rightsquigarrow (2)(3)\cdots(k)(k+1) \end{array} \right.$$

and this succession rule corresponds to combinatorial classes enumerated by Catalan numbers $C_d = \frac{1}{d+1}\binom{2d}{d}$ [2] .

Figure 1 shows the beginning of the generating tree associated with this ECO construction.

3 A first Gray code for $2d$ minimal permutations

Each $2d$ minimal permutation π is a node at level d in the generating tree (the root is at level 1) and it can be associated to a word (or a *code*), of length d, $w_1w_2 \ldots w_{d-1}w_d$, where each w_i is the label at level i in the path from the root to π. For example, in Figure 1 the code associated to the permutation 2143 is 22, while the one associated to the permutation 51627384 is 2345. In [4] an algorithm for generating codes for Catalan

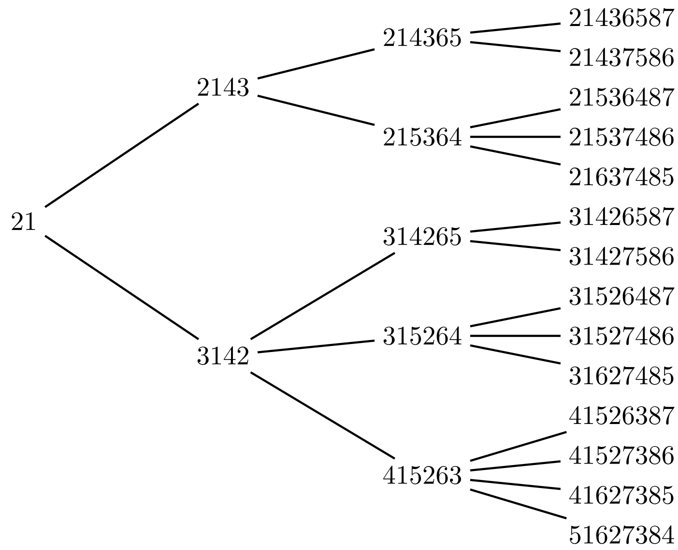

Figure 1: The first four levels of the generating tree associated with the ECO construction of $2d$ minimal permutation

structures, that is combinatorial objects enumerated by Catalan numbers, was defined such that two consecutive codes differ only by one digit. This generating algorithm is based on the so called *shifted production*. The labels of the sons of a node are not visited in the same order as they were in the production of the succession rule for Catalan numbers, where the list of the successors of a label (k) is $\langle 2, 3, \ldots, k, k+1 \rangle$. In this case the labels are *shifted* in such a way that the list $s(k, i)$, of the labels of the sons of a node with label k such that the first son has label i, is:

$$\begin{cases} s(k,2) = \langle 2, k+1, k, k-1, \ldots 4, 3 \rangle \\ s(k,i) = \langle i, i+1, \ldots, k-1, k, k+1, 2, 3, \ldots, i-1 \rangle, \ i \neq 2 \ . \end{cases}$$

The computation tree of the algorithm for codes of length 4 is shown in Figure 2; each leaf represents a unique code that is obtained by tracing the path from the root to the leaf. The list $\mathcal{L}_4$ of all the codes of length 4 is

$$\mathcal{L}_4 = \langle 2222, 2223, 2233, 2234, 2232, 2332, 2334, 2333, 2343, 2344, 2345, 2342, 2322, 2323 \rangle$$

Remark that the first and last children of each node are always 2 or 3. In [4] the following theorem is proved.

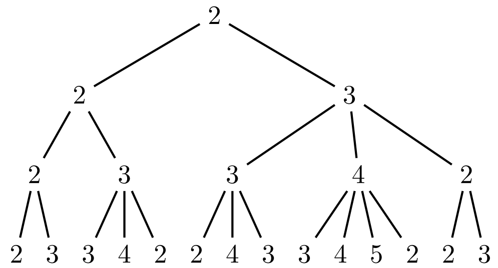

Figure 2: Computation tree to generate $\mathcal{L}_4$ in Gray code order

THEOREM 3.1 *Two consecutive elements of the list $\mathcal{L}_n$ differ only by one digit.*

That is to say that, using the above shifted productions, the algorithm generates the codes for $2d$ minimal permutations in Gray code order.

The basic idea behind the algorithm in [4] was generalized, in some sense, in [8], where an algorithm for generating *reflectable languages* in Gray code order is developed.

DEFINITION 3.2 A language L over the alphabet Σ is said to be *reflectable* if for every $i > 1$ there exist two distinct characters x_i and y_i in Σ such that if $w_1w_2 \dots w_{i-1}$ is a prefix of a word in L then both $w_1w_2 \dots w_{i-1}x_i$ and $w_1w_2 \dots w_{i-1}y_i$ are also prefixes of words in L.

Since the codes for the $2d$ minimal permutations correspond to a reflectable language where $x_i = 2$ and $y_i = 3\ \forall i$, the algorithm developed in [8] can be applied to generate the codes in Gray code order. It is easy to prove that both the algorithms in [4] and in [8] generate the same lists of codes.
Two consecutive codes differ only by one digit but this property does not hold for the corresponding $2d$ minimal permutations.

PROPOSITION 3.3 *Two words of the Gray code, differing by one digit, correspond to two $2d$ minimal permutations which differ by d elements at most.*

Proof. We can distinguish two cases.

Two codes differ in the last digit. Let us consider the two codes $w_1w_2 \dots w_{d-1}w_d$ and $w_1w_2 \dots w_{d-1}z_d$ and let σ, π be the two corresponding $2d$ minimal permutations. Following the ECO construction given in Section 2 we have $\sigma_{2d} = 2d - w_d + 1$ and $\pi_{2d} = 2d - z_d + 1$. Since the codes differ in the last digit, then σ and π are generated from the same minimal

Codes	Minimal permutations	Codes	Minimal permutations
2222	21436587	2333	$31\underline{5}2748\underline{6}$
2223	$2143\underline{7}58\underline{6}$	2343	$\underline{4}15273\underline{8}6$
2233	$21\underline{5}37\underline{4}86$	2344	$41\underline{6}27385\underline{5}$
2234	$21\underline{6}37485\underline{5}$	2345	$\underline{5}162738\underline{4}$
2232	$21\underline{5}3\underline{6}48\underline{7}$	2342	$\underline{4}1\underline{5}2\underline{6}38\underline{7}$
2332	$\underline{3}15\underline{2}6487$	2322	$\underline{3}1\underline{4}26\underline{5}87$
2334	$31\underline{6}2\underline{7}48\underline{5}$	2323	$3142\underline{7}58\underline{6}$

Table 1: The codes of length 4 and the corresponding minimal permutations of size 8 with 4 descents

permutation τ of size $2d-2$ by adding the value $2d$ in position $2d-1$ and then the values σ_{2d} and π_{2d}, respectively, in position $2d$.

Let $z_d = w_d + h$, with $h > 0$; then $\pi_{2d} = 2d - w_d - h + 1 = \sigma_{2d} - h$. In σ the elements j of τ with $j \geq \sigma_{2d}$ are turned into $j+1$, while in π those $\geq \pi_{2d}$ are turned into $j+1$, that is to say the elements that are greater than or equal to $(\sigma_{2d} - h)$. Therefore σ e π differ by $h+1$ elements (the last one and the h additional elements increased in π).

If $z_d = w_d - h$ $(h > 0)$, then $\pi_{2d} = \sigma_{2d} + h$. Therefore σ e π differ again by $h+1$ elements.

It is well-known that $2 \leq w_d \leq d+1$, hence $h \leq (d-1)$. Therefore the two minimal permutations differ by d elements at most.

Two codes differ by two digits which are not the last ones. Let us consider the two codes $w_1w_2\dots w_{j-1}w_jw_{j+1}\dots w_d$ and $w_1w_2\dots w_{j-1}$ $z_jw_{j+1}\dots w_d$. We can apply the same argument of the previous case to the minimal permutations σ e π of size $2j$ corresponding to codes $w_1w_2\dots w_{j-1}w_j$ and $w_1w_2\dots w_{j-1}z_j$, respectively. Indeed, the appending of the digit w_{j+1} to the code implies to add the pair $(2j+2, 2j-w_{j+1}+3)$ on the right of σ (resp. π); likewise for the digits $w_{j+2},\dots,w_d$. Hence the two minimal permutations corresponding to codes $w_1w_2\dots w_{j-1}w_jw_{j+1}\dots w_d$ and $w_1w_2\dots w_{j-1}z_jw_{j+1}\dots w_d$ are equal in positions $2j+1, 2j+2, \dots, 2d$. Again the two minimal permutations differ by $h+1$ elements, with $h = |w_j - z_j|$, that is by j elements at most. □

EXAMPLE 3.4 Table 1 shows the list $\mathcal{L}_4$ and the corresponding minimal permutations of size 8 with 4 descents. The underlined elements in a minimal permutation denote the differences compared to the preceding one.

4 An improved Gray code

Two $2d$ minimal permutations differ by d elements, at most, because the difference between the two different digits is not constant and it is d–1, at most. Therefore, the difference between the two different digits in consecutive codes sets the number of different elements in the corresponding minimal permutations. A possible proposal is to modify the order of elements in the shifted list of sons $s(k, i)$.
Note that:

- the proof of Theorem 3.1 given in [4] is based on the fact that the first element of $s(k, i)$ is always i,
- in the case of Catalan structures, only $s(k, 2)$ and $s(k, 3)$ occur in the construction of list $\mathcal{L}_n$,
- Theorem 3.1 still holds if the list of sons $s(k, i)$ is altered without modifying the first elements, which will be always 2 or 3,
- in the algorithm developed in [8] the special characters x_i and y_i are used as the first and last children of each node at level $i - 1$ and the order of the other children does not matter.

The new shifted productions are so defined:

- *k even*

$$s(k,2) = \langle \overbrace{2, 4, \ldots k-2, k}^{even}, \underbrace{k+1, k-1, \ldots 5, 3}_{odd} \rangle$$

$$s(k,3) = \langle \underbrace{3, 5, \ldots k-1, k+1}_{odd}, \overbrace{k, k-2, \ldots 4, 2}^{even} \rangle$$

- *k odd*

$$s(k,2) = \langle \overbrace{2, 4, \ldots k-1, k+1}^{even}, \underbrace{k, k-2, \ldots 5, 3}_{odd} \rangle$$

$$s(k,3) = \langle \underbrace{3, 5, \ldots k-2, k}_{odd}, \overbrace{k+1, k-1, \ldots 4, 2}^{even} \rangle$$

EXAMPLE 4.1 Applying the above definitions of $s(k, 2)$ and $s(k, 3)$ we have:

$$s(2,2) = \langle 2,3 \rangle \qquad s(2,3) = \langle 3,2 \rangle$$
$$s(3,2) = \langle 2,4,3 \rangle \qquad s(3,3) = \langle 3,4,2 \rangle$$
$$s(4,2) = \langle 2,4,5,3 \rangle \qquad s(4,3) = \langle 3,5,4,2 \rangle$$

The new definitions of $s(k,i)$ can be directly used in the algorithm in [4], while the procedure GrayCode(t) given in [8] must be updated as shown in Figure 3. In particular, in the **for each** loop the values of z must be taken in the order established by the shifted production and the check whether or not $w_1 \dots w_{t-1}z$ is a prefix of some word in L_n is dropped (we obtain a word in L_n for each z).

```
procedure GrayCode(t)
begin
if (t > n) then Process(w)
else r := w_t
     GrayCode(t + 1)

     for each z ∈ s(w_{t-1}, w_t) - {x_t, y_t}
               w_t := z
               GrayCode(t + 1)

     if (r = x_t) then w_t := y_t
     else w_t := x_t
     GrayCode(t + 1)
end
```

Figure 3: New version of the algorithm GrayCode(t)

EXAMPLE 4.2 By using the new definitions of $s(k,i)$, the computation tree of both the algorithms in [4] and in [8] is given in Figure 4.

The different digits in the two consecutive codes obtained differ by 2, at most. Therefore we can state the following proposition.

PROPOSITION 4.3 *The minimal permutations corresponding to two consecutive codes obtained by means of the new definitions of $s(k,i)$ differ in 3 elements, at most, independently of their size.*

In [4] the authors present a non-recursive algorithm to generate any code of length d based on the algorithm of Walsh [9] and they prove that their algorithm is $O(1)$ worst-case time per word. It is easy to prove that the introduction of the new definitions of $s(k,i)$ does not change the complexity of the procedure given in [4].

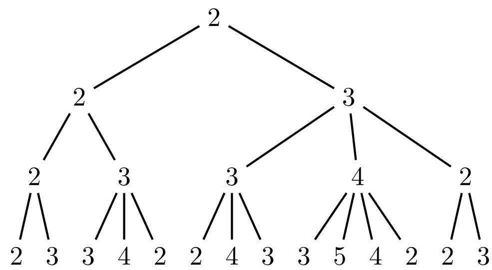

Figure 4: Computation tree to generate $\mathcal{L}_4$ with the new shifted productions

In [8] the following theorem is proved.

THEOREM 4.4 *The algorithm* GrayCode(t) *runs in a constant amortized time if the following two conditions hold:*

1. *checking whether or not* $w_1 \dots w_{t-1}z$ *is a prefix of some word in* L_n *takes O(1) time and*

2. *each internal node in the computation tree has* $\Theta(|\ \Sigma\ |)$ *children.*

In the new version of the algorithm GrayCode(t) the check in condition 1. is dropped and every internal node with value k has k children. Then, the new version of the algorithm GrayCode(t) runs in a constant amortized time, too.

References

[1] C. BANDERIER, M. BOUSQUET-MÉLOU, A. DENISE, P. FLAJOLET, D. GARDY, D. GOUYOU-BEAUCHAMPS, *Generating functions for generating trees*, Discrete Math., 246 (2002) 29–55.

[2] E. BARCUCCI, A. DEL LUNGO, E. PERGOLA, E. PINZANI, *ECO: a methodology for the enumeration of combinatorial objects*, J. Difference Equ. Appl., 5 (1999) 435–490.

[3] A. BERNINI, L. FERRARI, R. PINZANI, *Enumerating permutations avoiding three Babson-Steingrímsson patterns*, Annals of Combinatorics, 9 (2005) 137–162.

[4] A. Bernini, E. Grazzini, E. Pergola, R. Pinzani, *A general exhaustive generation algorithm for Gray structures*, Acta Informatica, 44 (2007) 361–376.

[5] M. Bouvel, E. Pergola, *Posets and Permutations in the Duplicated-Loss Model: Minimal Permutations with d Descents*, Theoret. Comput. Sci., 411 (2010) 2487–2501.

[6] M. Bouvel, D. Rossin, *A variant of the tandem duplication - random loss model of genome rearrangement*, Theoret. Comput. Sci, 410 (2009) 847–858.

[7] F. R. K. Chung, R. L. Graham, V. E. Hoggat, M. Kleiman, *The number of Baxter permutations*, J. Combin. Theory Ser. A, 24 (1978) 382–394.

[8] Y. Li, J. Sawada, *Gray codes for reflectable languages*, Information Processing Letters, 109 (2009) 296–300.

[9] T. Walsh, *Generating Gray Codes in O(1) worst-case time per word*, LNCS 2731 (2003) 73–88.