Gray code for permutations with a fixed number of left-to-right minima

Jean-Luc BARIL
LE2I UMR-CNRS 5158, Université de Bourgogne
B.P. 47 870, 21078 DIJON-Cedex France
e-mail: barjl@u-bourgogne.fr

January 13, 2009

Abstract

In [1], the author provided a Gray code for the set of n-length permutations with a given number of left-to-right minima in *inversion array representation*. In this paper, we give the first Gray code for the set of n-length permutations with a given number of left-to-right minima in *one-line representation*. In this code, each permutation is transformed into its successor by a product with a transposition or a cycle of length three. Also a generating algorithm for this code is given.

Keywords: Stirling numbers, permutations, left-to-right minima, Gray codes, generating algorithms.

1 Introduction

Let S_n be the set of all permutations of length n ($n \geq 1$). We represent permutations in one-line notation, i.e. if $i_1, i_2, \ldots, i_n$ are n distinct values in $[n] = \{1, 2, \ldots, n\}$, we denote the permutation $\sigma \in S_n$ by the sequence $(i_1, i_2, \ldots, i_n)$ if $\sigma(k) = i_k$ for $1 \leq k \leq n$. Moreover, if $\gamma = (\gamma(1), \gamma(2), \ldots, \gamma(n))$ is an n-length permutation then the *composition* (*or product*) $\gamma \cdot \sigma$ is the permutation $(\gamma(\sigma(1)), \gamma(\sigma(2)), \ldots, \gamma(\sigma(n)))$. In S_n, a *k-cycle* $\sigma = \langle i_1, i_2, \ldots, i_k\rangle$ is an n-length permutation verifying $\sigma(i_1) = i_2$, $\sigma(i_2) = i_3, \ldots, \sigma(i_{k-1}) = i_k$, $\sigma(i_k) = i_1$ and $\sigma(j) = j$ for $j \in [n]\backslash\{i_1, \ldots, i_k\}$; in particular, a *transposition* is a 2-cycle ($k = 2$). Now, let $\sigma \in S_n$. We say that $\sigma(i)$ is a *left-to-right minimum* of σ if $\sigma(i) < \sigma(j)$ for $j < i$. For instance, permutation 231546 has two left-to-right minima (2 and 1) and 654321 has six left-to-right minima. For

$1 \leq k \leq n$, we denote by $S_{n,k}$ the set of all n-length permutations with exactly k left-to-right minima. For instance, we have $S_{3,1} = \{123, 132\}$, $S_{3,2} = \{213, 231, 312\}$ and $S_{3,3} = \{321\}$. Obviously, $\{S_{n,k}\}_{1 \leq k \leq n}$ forms a partition for S_n and it is well known (see for instance [14], p. 20) that the cardinality of $S_{n,k}$ is given by the signless Stirling numbers of the first kind $s(n, k)$ satisfying:

$$s(n,k) = (n-1) \cdot s(n-1,k) + s(n-1,k-1) \qquad (1)$$

with the initial conditions $s(n, k) = 0$ if $n \leq 0$ or $k \leq 0$, except $s(0,0) = 1$. Notice that $s(n, k)$ also enumerates the n-length permutations with exactly k cycles. [1, 14, 18].

A *Gray code* is a family $\{\mathcal{L}_n\}_{n \geq 0}$ of lists of n-length sequences such that in each list the Hamming distance between any two consecutive sequences (i.e. the number of positions in which they differ) is bounded by a constant independently of the length of the sequences. If this constant is minimal then the code is called *optimal.* A Gray code is *cyclic* if the Hamming distance between the first and the last sequence is also bounded by this constant. There are many studies on generating permutations: in any order (or lex order) [5, 13], in Gray code order [4, 15]. Some other results are published for restricted permutations [3, 6, 8, 10, 12], derangements [2, 7], with a fixed number of cycles [1], involutions and fixed-point free involutions [17] or their generalizations (multiset permutations [16]). At [9] is given an implementation of U. Taylor and F. Ruskey [11] generating algorithm for n-length permutations with k left-to-right minima. However the generating order is neither lexicographic nor Gray code order.

In this paper, we provide the first Gray code for permutations (in one-line notation) with a fixed number of left-to-right minima. In this code two successive permutations differ in at most three positions (or equivalently, by a product with a 2 or 3-cycle). This code is optimal; indeed, it is easy to check that the set $S_{4,3} = \{3214, 3241, 3421, 4231, 4312, 4213\}$ can not be list such that two consecutive elements differ in at most two positions. This means there does not exist any Gray code for the complete family $\{S_{n,k}\}_{n \geq k \geq 1}$ such that two consecutive permutations in a list $S_{n,k}$ differ in at most two positions. However by considering the set $I = \mathbb{N} \times \{1\}$, we can obtain a Gray code for $\{S_{n,k}\}_{(n,k) \in I}$ such that the Hamming distance between any two consecutive permutations is two (this case will not be presented here).

Notice that in [1], the author gave a Gray code for n-length permutations with exactly k cycles and deduces a Gray code for n-length permutations with exactly k left-to-right minima in *inversion array representation* but not in *one-line notation.* Since there is not known gray code-preserving bijection between these representations it is of interest to construct directly a Gray code for permutations with a fixed number of left-to-right minima in the natural representation, *i.e.* in one-line notation. This is the main motivation of our work. In order to obtain this result, we first give two combinatorial interpretations of the recursive relation (1) which allows us to obtain in Section 3 two recursive definition- (2) and (3)- for the set $S_{n,k}$; then, by considering them in terms of lists, we show how one can recursively construct a Gray code for $S_{n,k}$. Finally we give a constant amortized time algorithm for generating our code.

2 Preliminaries

In this section we recursively construct $S_{n,k}$ from $S_{n-1,k}$ and $S_{n-1,k-1}$ which allows us to obtain a constructive proof of the enumerating relation (1). We also provide two lemmas, crucial in our construction of the code.

Let $\gamma \in S_{n-1,k}$ be an $(n-1)$-length permutation with k left-to-right minima, $n-1 \geq k \geq 1$; let also i be an integer, $1 \leq i \leq n-1$. If we denote by σ the permutation in S_n obtained from γ by inserting the entry n in the i-th position (i.e. between $\gamma(i)$ and $\gamma(i+1)$), then σ is an n-length permutation with k left-to-right minima.

Similarly, if $\gamma \in S_{n-1,k-1}$ is an $(n-1)$-length permutation with $(k-1)$ left-to-right minima, $n \geq k \geq 2$, and if σ denotes the permutation in S_n obtained from γ by appending n before the first position, then σ is an n-length permutation with k left-to-right minima. Moreover, each permutation in $S_{n,k}$, $n \geq 2$, can uniquely be obtained by one of these two constructions.

The functions ϕ_n and ψ_n defined below induce a bijection between $S_{n-1,k-1} \cup [n-1] \times S_{n-1,k}$ and $S_{n,k}$ such that its restriction to $S_{n-1,k-1}$ is ϕ_n and its restriction to $[n-1] \times S_{n-1,k}$ is ψ_n.

Definition 1 *1. For $1 \leq k < n$, an integer $i \in [n-1]$ and a permutation $\gamma \in S_{n-1,k}$, we define an n-length permutation $\sigma = \psi_n(i,\gamma)$ by*

$$\sigma(j) = \begin{cases} \gamma(j) & \text{if } j \leq i \\ n & \text{if } j = i+1 \\ \gamma(j-1) & \text{otherwise.} \end{cases}$$

2. For $n \geq k \geq 2$ and a permutation $\gamma \in S_{n-1,k-1}$, we define an n-length permutation $\sigma = \phi_n(\gamma)$ by

$$\sigma(j) = \begin{cases} n & \text{if } j = 1 \\ \gamma(j-1) & \text{otherwise.} \end{cases}$$

On the other hand, we have another bijection between $S_{n-1,k-1} \cup [n]\backslash\{1\} \times S_{n-1,k}$ and $S_{n,k}$ by replacing ϕ_n and ψ_n with the two functions ϕ'_n and ψ'_n given in Definition 2 below. Indeed, let us consider $\sigma \in S_{n,k}$, $n \geq 2$; if $\sigma(n) = 1$ then $\gamma = (\sigma(1)-1)(\sigma(2)-1)\ldots(\sigma(n-1)-1)$ belongs to $S_{n-1,k-1}$ and $\phi'(\gamma) = \sigma$; if $\sigma(n) = i$, $i \neq 1$, then we define γ in $S_{n-1,k}$ as the permutation obtained from σ by deleting the last entry i and by decreasing by one each entry greater than i, *i.e.*, $\sigma = \psi'(i,\gamma)$. Also, each permutation σ in $S_{n,k}$ can be uniquely obtained from one of these two constructions.

Definition 2 *1. For $1 \leq k < n$, an integer $i \in [n]\backslash\{1\}$ and a permutation $\gamma \in S_{n-1,k}$, we define an n-length permutation $\sigma = \psi'_n(i,\gamma)$ by*

$$\sigma(j) = \begin{cases} \gamma(j) & \text{if } \gamma(j) \leq i-1 \\ i & \text{if } j = n \\ \gamma(j)+1 & \text{otherwise.} \end{cases}$$

2. For $n \geq k \geq 2$ and a permutation $\gamma \in S_{n-1,k-1}$, we define an n-length permutation $\sigma = \phi'_n(\gamma)$ by

$$\sigma(j) = \begin{cases} 1 & \text{if } j = n \\ \gamma(j)+1 & \text{otherwise.} \end{cases}$$

In the following, we will omit the subscript n for the functions ϕ_n, ψ_n, ϕ'_n and ψ'_n, and it should be clear from context. Also, we will extend the functions ϕ, ψ, ϕ' and ψ' in a natural way to sets and lists of permutations. Moreover, if $\mathcal{S}$ is a list then $\overline{\mathcal{S}}$ is the list obtained by reversing $\mathcal{S}$, and $first(\mathcal{S})$ ($last(\mathcal{S})$ respectively) is the first (last respectively) element of the list, and obviously $first(\mathcal{S}) = last\left(\overline{\mathcal{S}}\right)$ and $first\left(\overline{\mathcal{S}}\right) = last(\mathcal{S})$; $\mathcal{S}^{(i)}$ is the list $\mathcal{S}$ if i is even, and $\overline{\mathcal{S}}$ if i is odd; if $\mathcal{S}_1$ and $\mathcal{S}_2$ are two lists, then $\mathcal{S}_1 \circ \mathcal{S}_2$ is their concatenation, and generally $\bigcirc_{i=1}^{m} \mathcal{S}_i$ is the list $\mathcal{S}_1 \circ \mathcal{S}_2 \circ \ldots \circ \mathcal{S}_m$. Similarly, $\bigcirc_{i=m}^{1} \mathcal{S}_i$ is the list $\mathcal{S}_m \circ \mathcal{S}_{m-1} \circ \ldots \circ \mathcal{S}_1$. Finally, for $i \in [n-1]$ and $\mathcal{S}$ a list of $(n-1)$-length permutations we have $\psi(i,\overline{\mathcal{S}}) = \overline{\psi(i,\mathcal{S})}$, $\psi(i, first(\mathcal{S})) = first(\psi(i,\mathcal{S}))$, and $\psi(i, last(\mathcal{S})) = last(\psi(i,\mathcal{S}))$. We obtain similar results for the function ψ' and for ϕ, ϕ' if we do not consider the parameter i.

In the sequel, $\mathcal{S}_{n,k}$ will denote our Gray code for the set $S_{n,k}$.

Here we give some elementary results which are crucial in the construction of our Gray code.

Lemma 1 *Let γ be an $(n-1)$-length permutation, if $n \geq 3$ and $1 \leq i \leq n-2$, then $\psi(i,\gamma) = \psi(i+1,\gamma) \cdot \langle i+1, i+2 \rangle$.*

Lemma 2 *Let γ be an $(n-1)$-length permutation, if $n \geq 3$ and $1 \leq i \leq n-3$, then $\psi(i,\gamma) = \psi(i+2,\gamma) \cdot \langle i+1, i+3, i+2 \rangle$.*

These two Lemmas can also be written for ψ' as follows:

Lemma 3 *Let γ be an $(n-1)$-length permutation, if $n \geq 3$ and $2 \leq i \leq n-1$, then $\psi'(i,\gamma) = \langle i, i+1 \rangle \cdot \psi'(i+1,\gamma)$.*

Lemma 4 *Let γ be an $(n-1)$-length permutation, if $n \geq 3$ and $2 \leq i \leq n-2$, then $\psi'(i,\gamma) = \langle i, i+1, i+2 \rangle \cdot \psi'(i+2,\gamma)$.*

3 The Gray code

From the remarks before Definition 1 results that the set $S_{n,k}$ can be written as:

$$S_{n,k} = \bigcup_{i=1}^{n-1} \psi(i, S_{n-1,k}) \cup \phi(S_{n-1,k-1}) \tag{2}$$

with $\phi(S_{n,0})$ and $\psi(i, S_{n,n+1})$ empty. Notice that if we consider ϕ' and ψ' the previous relation becomes

$$S_{n,k} = \bigcup_{i=2}^{n} \psi'(i, S_{n-1,k}) \cup \phi'(S_{n-1,k-1}). \tag{3}$$

If $\mathcal{S}$ is a list of permutations where any two consecutive permutations differ in p positions ($p \geq 1$) then so is the image of $\mathcal{S}$ by ψ, ψ', ϕ or ϕ'. Therefore, it is natural to look for a Gray code for the set $S_{n,k}$ of the form

$$\mathcal{S}_1 \circ \mathcal{S}_2 \circ \ldots \circ \mathcal{S}_\ell \circ \mathcal{T} \circ \mathcal{S}_{\ell+1} \circ \ldots \circ \mathcal{S}_{n-1} \tag{4}$$

where $\mathcal{T}$ is the list $\phi(\mathcal{S}_{n-1,k-1})$ or its reverse, and $\mathcal{S}_i$ is $\psi(j, \mathcal{S}_{n-1,k})$ or its reverse, for some j. Notice that (4) is an ordered counterpart of (2) and we have a similar result by considering relation (3).

In order to construct our Gray code, we distinguish five cases (*i*) $k = 1 \leq n$, (*ii*) $2 \leq k = n$, (*iii*) $2 \leq k = n-1$, (*iv*) $2 \leq k = n-2$ and (*v*) the other cases. For each case we give a recursive definition for an ordered list $\mathcal{S}_{n,k}$ of the set $S_{n,k}$, and we provide its first element $f_{n,k}$ and last element $\ell_{n,k}$. $\mathcal{S}_{n,k}$ is the concatenation of n lists as in (4) and we prove that it is a Gray code by showing that there is a 'smooth' transition between successive sublists. That is, the last permutation of a sublist and the first one of the next sublist differ in at most three positions. By the remark in introduction, the Gray code will be optimal.

3.1 The case $k = n$, $n \geq 1$

Obviously, $\mathcal{S}_{n,n}$, $n \geq 1$, contains only one element $(n, n-1, \ldots, 2, 1)$,
(a)
and in this case, there is nothing to do.

3.2 The case $k = 1$, $n \geq 2$

For $n \geq 2$ we define the relations (b):
• For $n = 2$, $\mathcal{S}_{n,1} = (1, 2)$.
• For $n = 2m \geq 2$,
$\mathcal{S}_{n,1} = \psi(n-1, \mathcal{S}_{n-1,1}) \circ \bigcirc_{i=m-1}^{1} \psi(2i-1, \mathcal{S}_{n-1,1})^{(m+i-1)} \circ$
$\bigcirc_{i=1}^{m-1} \psi(2i, \mathcal{S}_{n-1,1})^{(m+i)}$.
• For $n = 2m+1 \geq 3$,
$\mathcal{S}_{n,1} = \psi(n-1, \mathcal{S}_{n-1,1}) \circ \bigcirc_{i=m-1}^{1} \psi(2i, \mathcal{S}_{n-1,1})^{(m+i)} \circ$
$\bigcirc_{i=1}^{m} \psi(2i-1, \mathcal{S}_{n-1,1})^{(m-1+i)}$.

Remark that the function ϕ does not appear in these relations. See Figure 1 for an illustration of this code.

Figure 1: The relations (b) for $n = 8$ and $n = 9$. Each point i corresponds to the list $\psi(i, \mathcal{S}_{n-1,1})$ and each encircled point i represents the list $\overline{\psi(i, \mathcal{S}_{n-1,1})}$, *i.e.*, the reverse list of $\psi(i, \mathcal{S}_{n-1,1})$. For instance, if $n = 8$ then $\mathcal{S}_{8,1} = \psi(7, \mathcal{S}_{7,1}) \circ \psi(5, \mathcal{S}_{7,1}) \circ \overline{\psi(3, \mathcal{S}_{7,1})} \circ \psi(1, \mathcal{S}_{7,1}) \circ \overline{\psi(2, \mathcal{S}_{7,1})} \circ \psi(4, \mathcal{S}_{7,1}) \circ \overline{\psi(6, \mathcal{S}_{7,1})}$.

The lemma below gives the first and the last permutations of the list $\mathcal{S}_{n,1}$.

Lemma 5 *If $n \geq 3$ then*

1. $f_{n,1} = (1, 2, 3, \ldots, n-2, n-1, n)$
2. $\ell_{n,1} = (1, 2, 3, \ldots, n-2, n, n-1)$.

Proof. This holds if $n = 2$. For $n \geq 2$, relations (b) give $f_{n,1} = \psi(n-1, f_{n-1,1})$ and $\ell_{n,1} = \overline{\psi(n-2, \ell_{n-1,1})} = \psi(n-2, f_{n-1,1})$. We obtain the results by induction.

Proposition 1 *In the list $\mathcal{S}_{n,1}$ defined by (b), $n \geq 2$, two consecutive permutation differ in at most three positions.*

Proof. The transitions between $\psi(1, \mathcal{S}_{n-1,1})$ and $\overline{\psi(2, \mathcal{S}_{n-1,1})}$ (or between $\overline{\psi(1, \mathcal{S}_{n-1,1})}$ and $\psi(2, \mathcal{S}_{n-1,1})$) are given by Lemma 3. The transitions between $\psi(i, \mathcal{S}_{n-1,1})$ and $\overline{\psi(i+2, \mathcal{S}_{n-1,1})}$ (or $\overline{\psi(i, \mathcal{S}_{n-1,1})}$ and $\psi(i+2, \mathcal{S}_{n-1,1})$) are given by Lemma 4. Finally, for n even $n \geq 3$, we obtain

$$\begin{aligned} last(\psi(n-1, \mathcal{S}_{n-1,1})) &= \psi(n-1, last(\mathcal{S}_{n-1,1})) \\ &= (1,2,3,\ldots,n-3,n-1,n-2,n) \\ &= (1,2,3,\ldots,n-3,n,n-2,n-1)\cdot\langle n-2,n\rangle \\ &= \psi(n-3, first(\mathcal{S}_{n-1,1}))\cdot\langle n-2,n\rangle \\ &= first(\psi(n-3, \mathcal{S}_{n-1,1}))\cdot\langle n-2,n\rangle \end{aligned}$$

which gives a smooth transition (product by a transposition) between the two first sublists when n is even. □

3.3 The case $k = n-1$, $n \geq 3$

We define the relations (c) (see Figure 2),

- For $n = 2m \geq 4$,

$$\mathcal{S}_{n,n-1} = \psi'(n, \mathcal{S}_{n-1,n-1}) \circ \phi'(\mathcal{S}_{n-1,n-2}) \circ \bigcirc_{i=m-1}^{1} \psi'(2i, \mathcal{S}_{n-1,n-1}) \circ \bigcirc_{i=2}^{m} \psi'(2i-1, \mathcal{S}_{n-1,n-1}).$$

- For $n = 2m+1 \geq 3$,

$$\mathcal{S}_{n,n-1} = \psi'(n, \mathcal{S}_{n-1,n-1}) \circ \phi'(\mathcal{S}_{n-1,n-2}) \circ \bigcirc_{i=m}^{2} \psi'(2i-1, \mathcal{S}_{n-1,n-1}) \circ \bigcirc_{i=1}^{m} \psi'(2i, \mathcal{S}_{n-1,n-1}).$$

Notice that these relations consider ψ' and ϕ' instead of ψ and ϕ.

Lemma 6 *If $n \geq 3$ then*

1. $f_{n,n-1} = (n-1, n-2, n-3, \ldots, 2, 1, n)$
2. $\ell_{n,n-1} = (n, n-2, n-3, \ldots, 2, 1, n-1)$.

Proof. By relations (c) we obtain by induction $f_{n,n-1} = \psi'(n, \mathcal{S}_{n-1,n-1}) = (n-1, n-2, \ldots, 2, 1, n)$ and $\ell_{n,n-1} = \overline{\psi'(n-1, \mathcal{S}_{n-1,n-1})} = (n, n-2, n-3, \ldots, 2, 1, n-1)$.

Figure 2: The relations (c) for $n = 8$ and $n = 9$. Each point $i \neq 0$ corresponds to the list $\psi'(i, \mathcal{S}_{n-1,n-1})$ and the point 0 represents the list $\phi'(\mathcal{S}_{n-1,n-2})$. For instance, if $n = 8$ then $\mathcal{S}_{8,7} = \psi'(8, \mathcal{S}_{7,7}) \circ \phi'(\mathcal{S}_{7,6}) \circ \psi'(6, \mathcal{S}_{7,7}) \circ \psi'(4, \mathcal{S}_{7,7}) \circ \psi'(2, \mathcal{S}_{7,7}) \circ \psi'(3, \mathcal{S}_{7,7}) \circ \psi'(5, \mathcal{S}_{7,7}) \circ \psi'(7, \mathcal{S}_{7,7})$.

Proposition 2 *In the list $\mathcal{S}_{n,n-1}$ defined by (c), $n \geq 3$, two consecutive permutations differ in at most three positions.*

Proof. Here, we just consider the transitions between sublists which are not considered by Lemma 3 and 4. It remains the transitions (1) $\psi'(n, \mathcal{S}_{n-1,n-1}) \circ \phi'(\mathcal{S}_{n-1,n-2})$ and (2) $\phi'(\mathcal{S}_{n-1,n-2}) \circ \psi'(n-2, \mathcal{S}_{n-1,n-1})$. For the case (1),

$$\begin{aligned} last(\psi'(n, \mathcal{S}_{n-1,n-1})) &= (n-1, n-2, \ldots, 3, 2, 1, n) \\ &= (n-1, n-2, \ldots, 3, 2, n, 1) \cdot \langle n-1, n \rangle \\ &= \phi'(f_{n-1,n-2}) \cdot \langle n-1, n \rangle \\ &= first(\phi'(\mathcal{S}_{n-1,n-2})) \cdot \langle n-1, n \rangle. \end{aligned}$$

For the case (2),

$$\begin{aligned} last(\phi'(\mathcal{S}_{n-1,n-2})) &= (n, n-2, n-3, \ldots, 3, 2, n-1, 1) \\ &= (n, n-1, n-3, \ldots, 3, 2, 1, n-2) \cdot \langle 2, n, n-1 \rangle \\ &= first(\psi'(n-2, \mathcal{S}_{n-1,n-1})) \cdot \langle 2, n, n-1 \rangle. \end{aligned}$$

□

3.4 The case $k = n - 2$, $n \geq 4$

We define the relations (d) (see Figure 3),

- For $n = 2m \geq 4$,

$$\mathcal{S}_{n,n-2} = \psi(n-1, \mathcal{S}_{n-1,n-2}) \circ \overline{\phi(\mathcal{S}_{n-1,n-3})} \circ \bigcirc_{i=m-1}^{1} \psi(2i-1, \mathcal{S}_{n-1,n-2})^{(m+i-1)} \circ \bigcirc_{i=1}^{m-1} \psi(2i, \mathcal{S}_{n-1,n-2})^{(m+i)}.$$

- For $n = 2m + 1 \geq 5$,

$$\mathcal{S}_{n,n-2} = \psi(n-1, \mathcal{S}_{n-1,n-2}) \circ \phi(\mathcal{S}_{n-1,n-3}) \circ \bigcirc_{i=m-1}^{1} \psi(2i, \mathcal{S}_{n-1,n-2})^{(m+i)} \circ \bigcirc_{i=1}^{m} \psi(2i-1, \mathcal{S}_{n-1,n-2})^{(m+1+i)}.$$

Lemma 7 *If $n \geq 4$ then*

Figure 3: The relations (d) for $n = 8$ and $n = 9$. Each point $i \neq 0$ corresponds to the list $\psi(i, \mathcal{S}_{n-1,n-2})$ and the point 0 is the list $\phi(\mathcal{S}_{n-1,n-3})$. When the point is encircled, we consider the inverse list. For instance, if $n = 8$ then $\mathcal{S}_{8,6} = \psi(7, \mathcal{S}_{7,6}) \circ \overline{\phi(\mathcal{S}_{7,5})} \circ \psi(5, \mathcal{S}_{7,6}) \circ \overline{\psi(3, \mathcal{S}_{7,6})} \circ \psi(1, \mathcal{S}_{7,6}) \circ \overline{\psi(2, \mathcal{S}_{7,6})} \circ \psi(4, \mathcal{S}_{7,6}) \circ \overline{\psi(6, \mathcal{S}_{7,6})}$.

1. $f_{n,n-2} = (n-2, n-3, \ldots, 2, 1, n-1, n)$

2. $\ell_{n,n-2} = (n-2, n-3, \ldots, 2, 1, n, n-1)$.

Proof. By induction and with relations (d) we obtain
$f_{n,n-2} = \psi(n-1, first(\mathcal{S}_{n-1,n-2})) = \psi(n-1, (n-2, n-3, \ldots, 2, 1, n-1)) = (n-2, n-3, \ldots, 2, 1, n-1, n)$ and
$\ell_{n,n-2} = \psi(n-2, first(\mathcal{S}_{n-1,n-2})) = \psi(n-2, (n-2, n-3, \ldots, 2, 1, n-1)) = (n-2, n-3, \ldots, 2, 1, n, n-1)$. □

Proposition 3 *In the list $\mathcal{S}_{n,n-2}$ defined by (d), $n \geq 4$, two consecutive permutations differ in at most three positions.*

Proof. After applying Lemma 1 and 2, it remains the transitions (1) $\psi(n-1, \mathcal{S}_{n-1,n-2}) \circ \phi(\mathcal{S}_{n-1,n-3})$, (2) $\psi(n-1, \mathcal{S}_{n-1,n-2}) \circ \overline{\phi(\mathcal{S}_{n-1,n-3})}$, (3) $\phi(\mathcal{S}_{n-1,n-3}) \circ \overline{\psi(n-3, \mathcal{S}_{n-1,n-2})}$ and (4) $\overline{\phi(\mathcal{S}_{n-1,n-3})} \circ \psi(n-3, \mathcal{S}_{n-1,n-2})$.
For the cases (1) and (2),

$$
\begin{aligned}
last(\psi(n-1, \mathcal{S}_{n-1,n-2})) &= (n-1, n-3, n-4, \ldots, 3, 2, 1, n-2, n) \\
&= (n, n-3, n-4, \ldots, 1, n-2, n-1) \cdot \langle 1, n \rangle \\
&= \phi(f_{n-1,n-3}) \cdot \langle 1, n \rangle \\
&= first(\phi(\mathcal{S}_{n-1,n-3})) \cdot \langle 1, n \rangle \\
&= last(\phi(\mathcal{S}_{n-1,n-3})) \cdot \langle 1, n-1, n \rangle.
\end{aligned}
$$

For the case (3),

$$
\begin{aligned}
last(\phi(\mathcal{S}_{n-1,n-3})) &= (n, n-3, n-4, \ldots, 2, 1, n-1, n-2) \\
&= (n-1, n-3, \ldots, 2, n, 1, n-2) \cdot \langle 1, n-2, n-1 \rangle \\
&= last(\psi(n-3, \mathcal{S}_{n-1,n-2})) \cdot \langle 1, n-2, n-1 \rangle.
\end{aligned}
$$

For the case (4),

$$
\begin{aligned}
first(\phi(\mathcal{S}_{n-1,n-3})) &= (n, n-3, n-4, \ldots, 2, 1, n-2, n-1) \\
&= (n-2, n-3, \ldots, 2, n, 1, n-1) \cdot \langle 1, n-2, n-1 \rangle \\
&= first(\psi(n-3, \mathcal{S}_{n-1,n-2})) \cdot \langle 1, n-2, n-1 \rangle.
\end{aligned}
$$

□

3.5 The case $2 \le k \le n-3$

If $2 \le k \le n-3$, we define relations (e) (see Figure 4)

- For $n = 2m$, k odd,

$$\mathcal{S}_{n,k} = \psi(n-1, \mathcal{S}_{n-1,k}) \circ \bigcirc_{i=m-1}^{1} \psi(2i-1, \mathcal{S}_{n-1,k})^{(m+i)} \circ \bigcirc_{i=1}^{(k-1)/2} \psi(2i, \mathcal{S}_{n-1,k})^{(m+i+1)} \circ \phi(\mathcal{S}_{n-1,k-1})^{m+(k-1)/2} \circ \bigcirc_{i=(k+1)/2}^{m-1} \psi(2i, \mathcal{S}_{n-1,k-1})^{(m+i)}.$$

- For $n = 2m$, k even,

$$\mathcal{S}_{n,k} = \psi(n-1, \mathcal{S}_{n-1,k}) \circ \bigcirc_{i=m-1}^{(k+2)/2} \psi(2i-1, \mathcal{S}_{n-1,k})^{(m+i)} \circ \phi(\mathcal{S}_{n-1,k-1})^{m+\frac{k}{2}} \circ \bigcirc_{i=k/2}^{1} \psi(2i-1, \mathcal{S}_{n-1,k})^{(m+i-1)} \circ \bigcirc_{i=1}^{m-1} \psi(2i, \mathcal{S}_{n-1,k-1})^{(m+i)}.$$

- For $n = 2m+1$, k odd,

$$\mathcal{S}_{n,k} = \psi(n-1, \mathcal{S}_{n-1,k}) \circ \bigcirc_{i=m-1}^{(k+1)/2} \psi(2i, \mathcal{S}_{n-1,k})^{(m+i-1)} \circ \phi(\mathcal{S}_{n-1,k-1})^{m+\frac{k+1}{2}} \circ \bigcirc_{i=(k-1)/2}^{1} \psi(2i, \mathcal{S}_{n-1,k})^{(m+i)} \circ \bigcirc_{i=1}^{m} \psi(2i-1, \mathcal{S}_{n-1,k-1})^{(m+i+1)}.$$

- For $n = 2m+1$, k even,

$$\mathcal{S}_{n,k} = \psi(n-1, \mathcal{S}_{n-1,k}) \circ \bigcirc_{i=m-1}^{1} \psi(2i, \mathcal{S}_{n-1,k})^{(m+i-1)} \circ \bigcirc_{i=1}^{(k-1)/2} \psi(2i-1, \mathcal{S}_{n-1,k})^{(m+i)} \circ \phi(\mathcal{S}_{n-1,k-1})^{m+(k+1)/2} \circ \bigcirc_{i=(k+1)/2}^{m} \psi(2i-1, \mathcal{S}_{n-1,k-1})^{(m+i+1)}.$$

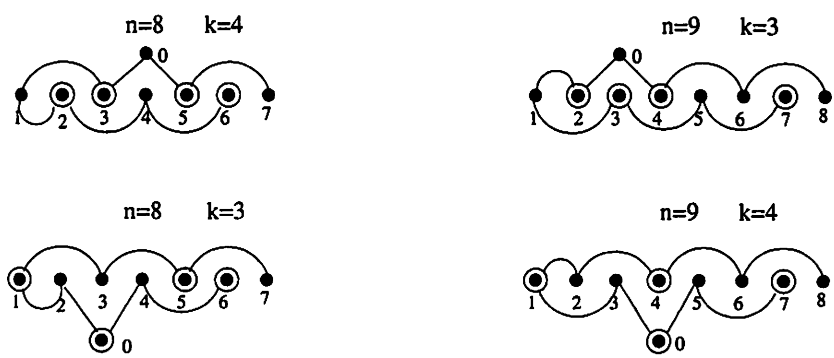

Figure 4: The relations (e) for $n = 8, 9$ and $k = 3, 4$. Each point $i \neq 0$ corresponds to the list $\psi(i, \mathcal{S}_{n-1,k})$ and the point 0 is the list $\phi(\mathcal{S}_{n-1,k-1})$. When the point is encircled, we consider the inverse list. For instance, if $n = 8$ and $k = 4$ then $\mathcal{S}_{8,4} = \psi(7, \mathcal{S}_{7,4}) \circ \overline{\psi(5, \mathcal{S}_{7,4})} \circ \phi(\mathcal{S}_{7,3}) \circ \overline{\psi(3, \mathcal{S}_{7,4})} \circ \psi(1, \mathcal{S}_{7,4}) \circ \overline{\psi(2, \mathcal{S}_{7,4})} \circ \psi(4, \mathcal{S}_{7,4}) \circ \overline{\psi(6, \mathcal{S}_{7,4})}$.

Lemma 8 *If* $2 \leq k \leq n-3$ *then*

1. $f_{n,k} = (k, k-1, k-2, \ldots, 2, 1, k+1, k+2, \ldots, n-2, n-1, n)$
2. $\ell_{n,k} = (k, k-1, k-2, \ldots, 2, 1, k+1, k+2, \ldots, n-2, n, n-1)$

Proof. We obtain the result with $f_{n,k} = first(\psi(n-1, \mathcal{S}_{n-1,k})) = \psi(n-1, f_{n-1,k})$ anchored by $f_{k,k} = (k, k-1, k-2, \ldots, 3, 2, 1)$. Similarly, $\ell_{n,k} = last(\psi(n-2, \overline{\mathcal{S}_{n-1,k}})) = \psi(n-2, f_{n-1,k})$ also anchored by $f_{k,k} = (k, k-1, k-2, \ldots, 3, 2, 1)$. □

Proposition 4 *In the list* $\mathcal{S}_{n,k}$ *defined by (e),* $2 \leq k \leq n-3$*, two consecutive permutations differ by at most three positions.*

Proof. After applying Lemma 1 and 2, it remains the transitions (1) $\psi(k-1, \mathcal{S}_{n-1,k}) \circ \phi(\mathcal{S}_{n-1,k-1})$, (2) $\overline{\phi(\mathcal{S}_{n-1,k-1})} \circ \psi(k+1, \mathcal{S}_{n-1,k})$, (3) $\overline{\psi(k-1, \mathcal{S}_{n-1,k})} \circ \phi(\mathcal{S}_{n-1,k-1})$, (4) $\phi(\mathcal{S}_{n-1,k-1}) \circ \overline{\psi(k+1, \mathcal{S}_{n-1,k})}$, (5) $\psi(n-1, \mathcal{S}_{n-1,k}) \circ \psi(n-3, \mathcal{S}_{n-1,k})$.

Case 1: $last(\psi(k-1, \mathcal{S}_{n-1,k})) =$
$= (k, k-1, \ldots, 2, n, 1, k+1, k+2, \ldots, n-1, n-2)$
$= (n, k-1, \ldots, 2, 1, k, k+1, k+2, \ldots, n-1, n-2) \cdot \langle 1, k+1, k \rangle$
$= first(\overline{\phi(\mathcal{S}_{n-1,k-1})}) \cdot \langle 1, k+1, k \rangle$.

Case 2: $first(\psi(k+1, \mathcal{S}_{n-1,k})) =$
$= (k, k-1, \ldots, 2, 1, k+1, n, k+2, \ldots, n-2, n-1)$
$= (n, k-1, \ldots, 2, 1, k, k+1, k+2, \ldots, n-2, n-1) \cdot \langle 1, k+1, k+2 \rangle$
$= last(\overline{\phi(\mathcal{S}_{n-1,k-1})}) \cdot \langle 1, k+1, k+2 \rangle$.

Case 3: $last(\overline{\psi(k-1, \mathcal{S}_{n-1,k})}) =$
$= (k, k-1, \ldots, 2, n, 1, k+1, k+2, \ldots, n-2, n-1)$
$= (n, k-1, \ldots, 2, 1, k, k+1, k+2, \ldots, n-2, n-1) \cdot \langle 1, k+1, k \rangle$
$= last(\phi(\mathcal{S}_{n-1,k-1})) \cdot \langle 1, k+1, k \rangle$.

Case 4: $first(\overline{\psi(k+1, \mathcal{S}_{n-1,k})}) =$
$= (k, k-1, \ldots, 2, 1, k+1, n, k+2, \ldots, n-1, n-2)$
$= (n, k-1, \ldots, 2, 1, k, k+1, k+2, \ldots, n-1, n-2) \cdot \langle 1, k+1, k+2 \rangle$
$= last(\phi(\mathcal{S}_{n-1,k-1})) \cdot \langle 1, k+1, k+2 \rangle$.

Case 5: $first(\psi(n-3, \mathcal{S}_{n-1,k})) =$
$= (k, k-1, \ldots, 2, 1, k+1, \ldots, n-3, n, n-2, n-1)$
$= (k, k-1, \ldots, 2, 1, k+1, \ldots, n-3, n-1, n, n-2) \cdot \langle n-2, n-1, n \rangle$
$= last(\psi(n-1, \mathcal{S}_{n-1,k})) \cdot \langle n-2, n-1, n \rangle$.

□

The family of lists $\{\mathcal{S}_{n,k}\}_{n \geq k \geq 1}$ is an optimal cyclic Gray code. See Table 1 for some examples. Moreover, let denote $\mathcal{S}_{n,k}^{-1}$ the list obtained

Table 1: The lists $\mathcal{S}_{3,k}$, $1 \le k \le 3$, and $\mathcal{S}_{4,k}$ for $1 \le k \le 4$. For instance, in $\mathcal{S}_{4,2}$ the sublists of relation (d), $\psi(3, \mathcal{S}_{3,2})$, $\overline{\phi(\mathcal{S}_{3,1})}$, $\psi(1, \mathcal{S}_{3,2})$, and $\overline{\psi(2, \mathcal{S}_{3,2})}$, are alternatively in bold-face and italic.

$\mathcal{S}_{3,1}$		$\mathcal{S}_{3,2}$		$\mathcal{S}_{3,3}$		$\mathcal{S}_{4,1}$		$\mathcal{S}_{4,2}$		$\mathcal{S}_{4,3}$		$\mathcal{S}_{4,4}$	
1	123	1	213	1	321	1	1234	1	2134	1	3214	1	4321
2	*132*	2	*231*			2	1324	2	2314	2	*3241*		
		3	312			3	*1423*	3	3124	3	*3421*		
						4	*1432*	4	*4132*	4	*4231*		
						5	1342	5	*4123*	5	4312		
						6	1243	6	2413	6	*4213*		
								7	2431				
								8	3412				
								9	*3142*				
								10	*2341*				
								11	*2143*				

from $\mathcal{S}_{n,k}$ by replacing each permutation in $\mathcal{S}_{n,k}$ by its group theoretical inverse. Thus, by a simple calculation, the family of lists $\{\mathcal{S}_{n,k}^{-1}\}_{n \ge k \ge 1}$ is also an optimal cyclic Gray code for the n-length permutations with k left-to-right minima; i.e. two successive permutations in a list $\mathcal{S}_{n,k}^{-1}$ differ by a product with a transposition or a 3-cycle.

4 Algorithmic considerations

In this part, we explain how the recursive definitions (a), (b), (c), (d) and (e) can be implemented into an efficient algorithm, i.e. in a constant amortized time (CAT) algorithm. Such algorithms already exist for derangements, permutations with k cycles or involutions [1, 2, 10, 12], so we will just give here the main difficulties to implement our one.

According to the relation (4) and the cases (a), (b), (c), (d) and (e), the procedure *gen_up*(n,k) given in Appendix produces iteratively the sublists $\mathcal{S}_i$ and $\mathcal{T}$ that are recursively generated. So, each call of this procedure fills up entries with indices in an active set $T \subset [n]$ associated with it, and for each recursive call $T = \{i_1, i_2, \ldots, i_k\}$ is replaced by an active subset T' such that $T = T' \cup \{i_j\}$ where $j \in [k]$. As we did it for the generating algorithm the Gray code for derangements [2], the current active set T is represented by four global variables: the integers *head*, *tail*, and the two arrays *succ* and *pred* defined as follows. If at a computational step $T = \{i_1, i_2, \ldots, i_k\}$, then we let $head = i_1$, $tail = i_k$, $succ[i_j] = i_{j+1}$ and $pred[i_j] = i_{j-1}$. So, before each recursive call of the procedure corresponding to the active set T', (1) the procedure *remove*(i_j) delete the index i_j in T, (2) we perform the recursive call relatively to $T' = T \setminus \{i_j\}$, (3) the procedure *append*(i_j) add the index i_j in T.

In our algorithm, we consider initially the active set $T = [n+1]$, and the procedure *gen_up*(n, k) generates permutations of $\mathcal{S}_{n,k}$ in an array p anchored by the first element of the list. The procedure *type*() prints out the current permutation of p.
Between any successive calls at least one update statement is performed (according to the relations (a), (b), (c), (d) and (e)), and after each update statement a new permutation is produced and printed out. The procedure generates at least two recursive calls or produces a permutation. Clearly, the time complexity of *gen_up* is proportional to the total number of recursive calls. Since each call produces at least one new element the time complexity is $\mathcal{O}(s_{n,k})$. A java implementation of our algorithm is avalaible at *http : //www.u-bourgogne.fr/LE2I/jl.baril/leftapplet.html.*

5 Appendix

The call of *gen_up(n,k)* generates the list $\mathcal{S}_{n,k}$. In order to produce $\overline{\mathcal{S}_{n,k}}$ we consider also the procedure *gen_down(n,k)* which has the same instructions of *gen_up(n,k)* in the reverse order. The notation *gen_up/down*$(n-1, k)$ means that we use *gen_up(n,k)* or *gen_down(n,k)* according to the sense of each sublist in the relations (a), (b), (c), (d) and (e). The notation $\langle i, j\rangle$ means that the current permutation p is composed (on the right) by the transposition $\langle i, j\rangle$.

```
public static void gen_up(int n, int k)
 { if(k==n) type();
   if(k==1)
      if(n==3) {type();<succ[head],succ[succ[head]]>;type();}
      else
      { run=pred[tail];remove(run);gen_up(n-1,k);append(run);
        <run,pred[run]>;<pred[run],pred[pred[run]]>;
        if(n mod 2==0)
         <pred[tail],pred[pred[tail]]>;
        for(i=n-3;i>=1;i=i-2)
        {run=pred[pred[run]];remove(run);gen_up/down(n-1,k);append(run);
         if(i>=3)
          {<run,pred[run]>;<pred[run],pred[pred[run]]>;}
         else <succ[head],succ[succ[head]]>;
        }
        if(i==-1) {i=2;run=succ[succ[head]];} else {i=1;run=succ[head];}
        for(int j=1;j<=n-2;j=j+2)
         {remove(run);gen_up/down(n-1,k);append(run);
          if(j+2<=n-2){<run,succ[run]>;<succ[run],succ[succ[run]]>;run=succ[succ[run]];}
         }
      }
   if(k==n-1)
        if(n==3){type();<pred[tail],pred[pred[tail]]>;type();
                     <pred[pred[pred[tail]]],pred[pred[tail]]>;
                     <pred[tail],pred[pred[tail]]>;type();}
        else
         { type();<pred[tail],pred[pred[tail]]>;run=pred[tail];
           remove(run);gen_up(n-1,k-1);append(run);
           run=succ[head];<pred[tail],pred[pred[tail]]>;<pred[tail],run>;type();
           run=head;
           for(i=n-4;i>=2;i=i-2)
            {run=succ[succ[run]];<run,succ[run]>;<run,pred[tail]>;type();}
           <pred[tail],pred[pred[pred[tail]]]>;type();
           if(i==0) {i=3;run=pred[pred[tail]];} else {i=2;run=pred[tail];}
           for(int j=i+2;j<=n-1;j=j+2)
             {run=pred[pred[run]];<run,pred[run]>;<run,pred[tail]>;type();}
         }
   if(k==n-2)
    {   run=pred[tail];
```

```
        remove(run);gen_up(n-1,k);append(run);
        if(n mod 2==0)
          {<head,pred[tail]>;<pred[tail],pred[pred[tail]]>;}
        else {<head,pred[tail]>;}
        run=head;
        remove(run);gen_up/down(n-1,k-1);append(run);
        <pred[pred[pred[tail]]],pred[pred[tail]]>;<head,pred[pred[pred[tail]]]>;
        run=pred[tail];
        for(i=n-3;i>=1;i=i-2)
         {run=pred[pred[run]];
          remove(run);gen_up/down(n-1,k);append(run);
          if(i>=3) {<run,pred[run]>;<pred[run],pred[pred[run]]>;}
          else {<succ[head],succ[succ[head]]>;}
         }
        if(i==-1) {i=2;run=succ[succ[head]];} else {i=1;run=succ[head];}
        for(int j=i;j<=n-2;j=j+2)
         {remove(run);gen_up/down(n-1,k);append(run);
          if(j+2<=n-2){<run,succ[run]>;<succ[run],succ[succ[run]]>;run=succ[succ[run]];}
         }
    }
   if(k!= n && k!= 1 && k!= n-1 && k!= n-2)
    {   run=pred[tail];
        remove(run);gen_up(n-1,k);append(run);
        if(n mod 2==0){<run,pred[run]>;<pred[run],pred[pred[run]]>;}
        else {<run,pred[pred[run]]>;}
        run=pred[tail];
        for(i=n-3;i>=1;i=i-2)
         { run=pred[pred[run]];
           remove(run);gen_up/down(n-1,k);append(run);
           if(i==k+1)
            {<pred[run],run>;<head,pred[run]>;run1=head;
             remove(run1);gen_up/down(n-1,k-1,run1,sens);append(run1);
             <head,pred[run]>;<pred[run],run>;
            }
           if(i>=3) {<run,pred[run]>;<pred[run],pred[pred[run]]>;}
           else {<succ[head],succ[succ[head]]>;}
         }
        if(i==-1) {i=2;run=succ[succ[head]];} else {i=1;run=succ[head];}
        for(int j=i;j<=n-2;j=j+2)
         { remove(run);gen_up/down(n-1,k);append(run);
           if(j==k-1)
            { <succ[run],run>;<head,succ[run]>;
              run1=head;
              remove(run1);gen_up/down(n-1,k-1);append(run1);
              <head,succ[run]>;<succ[run],run>;
            }
           if(j+2<=n-2){<run,succ[run]>;<succ[run],succ[succ[run]]>;run=succ[succ[run]];}
         }
    }
}
```

References

[1] J.L. Baril. Gray code for permutations with a fixed number of cycles. *Discrete Math.*, 307, 13(6): 1559–1571, 2007.

[2] J.L. Baril and V. Vajnovszki. Gray code for derangements. *Discrete Appl. Math.*, 140(1-3): 207–221, 2004.

[3] W.M.B. Dukes, M. F. Flanagan, T. Mansour and V. Vajnovszki. Combinatorial Gray codes for classes of pattern avoiding permutations. *Theoretical Computer Science*, 396: 35–49, 2008.

[4] S.M. Johnson. Generation of permutations by adjacent transpositions. *Mathematics of computation*, 17: 282–285, September 1963.

[5] D.E. Knuth. *The art of computer programming, combinatorial algorithms*, volume 4, fascicle 2, Generating All Tuples and Permutations, Addison Wesley, 2005.

[6] J.F. Korsh. Loopless generation of up-down permutations. *Discrete Math.*, 240(1-3): 97–122, 2001.

[7] J.F. Korsh. Constant time generation of derangements. *Information Process. Letters*, 90(4): 181–186, 2004.

[8] D. Roelants van Baronaigien. Constant time generation of involutions. *Congressus Numerantium*, 90: 87–96, 1992.

[9] F. Ruskey. http://www.theory.cs.uvic.ca/~cos/gen/perm.html.

[10] F. Ruskey and S. Effler. A CAT algorithm for generating permutations with a fixed number of inversions. *Inform. Process. Lett.*, 86(2): 107–112, 2003.

[11] F. Ruskey and U. Taylor. Fast generation of restricted classes of permutations. *Manuscript*, 1995.

[12] F. Ruskey and D. Roelants van Baronaigien. Generating permutations with given ups and downs. *Discrete Appl. Math.*, 36(1): 57–67, 1992.

[13] R. Sedgewick. Permutation generation methods. *Comput. Surveys*, 9(2): 137–164, 1977.

[14] R. Stanley. *Enumerative combinatorics*, volume 1. Cambridge University Press, Cambridge, England, 1997.

[15] H.F. Trotter. *PERM (Algorithm 115).* Communications of ACM, 5(8), 434–435, 1962.

[16] V. Vajnovszki. A loopless algorithm for generating the permutations of a multiset. *Theoretical Computer Science*, 307: 415–431, 2003.

[17] T. Walsh. Gray code for involutions. *J. Combin. Math. Combin. Comput.*, 36: 95–118, 2001.

[18] R.M. Wilson and J.H. van Lint. *A course in Combinatorics.* Cambridge University Press, 2002.