GRAY CODING CUBIC PLANAR MAPS

SERGEY AVGUSTINOVICH, SERGEY KITAEV, VLADIMIR N. POTAPOV,
AND VINCENT VAJNOVSZKI

ABSTRACT. The idea of (combinatorial) Gray codes is to list objects in question in such a way that two successive objects differ in some pre-specified small way. In this paper, we utilize β-description trees to cyclicly Gray code three classes of cubic planar maps, namely, bicubic planar maps, 3-connected cubic planar maps, and cubic non-separable planar maps.

1. INTRODUCTION

Gray codes. The problem of exhaustively listing the objects of a given class is important for several fields of science such as computer science, hardware and software, biology and (bio)chemistry. The idea of so-called *Gray codes* (or *combinatorial Gray codes*) is to list the objects in such a way that two successive objects differ in some pre-specified small way; in addition, if the last and first objects in the list differ in the same small way, then the Gray code is called cyclic. In [14] a general definition is given, where a Gray code is defined as an infinite set of wordlists with unbounded word-length such that the *Hamming distance* between any two successive words is bounded independently of the word-length (the Hamming distance is the number of positions in which the words differ).

Originally, a Gray code was used in a telegraph demonstrated by the French engineer Émile Baudot in 1878. However, these days we normally say "the Gray code" to refer to the *reflected binary code* introduced by Frank Gray in 1947 to list all binary words of length n. Much has been discovered and written about the Gray code (see for example [9] or [10, 5] for surveys) and it was used, for example, in *error corrections in digital communication* and in solving puzzles like *Tower of Hanoi puzzle*. On the other hand, the area of combinatorial Gray codes was popularized by Herbert Wilf in 1988-89 and since then such codes were found for many combinatorial structures, e.g. for *involutions* and *fixed-point free involutions*, *derangements* and certain classes of *pattern-avoiding permutations* (see [6, 10] and the references therein).

Existence of a (resp., cyclic) Gray code is often established via finding a *Hamiltonian path* (resp., *Hamiltonian cycle*) in a certain graph corresponding to the objects in question. In such a graph two vertices are connected by an edge if and only if the respective objects can follow each other in a Gray code. A Hamiltonian path (resp., Hamiltonian cycle) in a graph is a path (resp., cycle) in the graph that goes through each vertex exactly once.

Planar maps. A *planar map* is a connected graph, with loops and parallel edges allowed, embedded in the sphere with no edge-crossings, and considered up to orientation-preserving homeomorphism. A map has *vertices*, *edges*, and *faces*. A vertex is a point on the sphere. An edge which is not a loop is an open curve whose endpoints are its incident vertices. A loop is a closed curve which contains

---

its incident vertex. A face is a connected component of the complement of the underlying graph in the sphere, and it is homeomorphic to an open disc.

The maps we consider shall be *rooted*, meaning that a directed edge has been distinguished as the root. The *root face* is the face incident to the left side of the root as seen by an observer facing in the direction of the orientation of the root, so by continuing to walk around the boundary of the face, the observer traces this boundary in the counter-clockwise direction.

A planar map in which each vertex is of degree 3 is *cubic*; it is *bicubic* if, in addition, it is bipartite, that is, if its vertices can be colored using two colors, say, black and white, so that adjacent vertices are assigned different colors. A map is *k-connected* if there does not exist a set of $k-1$ vertices whose removal disconnects the map. 2-connected maps are also known as *non-separable maps*.

For brevity, we omit the word "planar" in the classes of planar maps considered in this paper.

Tutte [13, Chapter 10] founded the enumerative theory of planar maps in a series of papers in the 1960s (see [12] and the references in [3]). In particular, the number of bicubic maps and cubic non-separable maps on $2n$ vertices are, respectively,

$$\frac{3 \cdot 2^{n-1}(2n)!}{n!(n+2)!} \quad \text{and} \quad \frac{2^n(3n)!}{(n+1)!(2n+1)!}.$$

$\beta(a,b)$**-trees and planar maps.** A *valuated tree* is a rooted plane tree with non-negative integer labels on its vertices. A *description tree* introduced by Cori et al. in [2] is a valuated tree such that the label of each vertex v belongs to a set of values that depends only on the labels of v's sons according to a given rule. Description trees give a framework for recursively decomposing several families of planar maps. β-description trees, introduced next, are of interest in this paper.

A *plane tree* is a tree embedded in the plane as a map.

Definition 1. A $\beta(a,b)$-tree is a rooted plane tree whose vertices are labeled with non-negative integers such that

(1) leaves have label a;
(2) the label of the root is the sum of its children's labels;
(3) the label of any other vertex is at least a and at most b plus the sum of its children's labels.

It was shown in [2, 3] that the following objects are in one-to-one correspondence:
- $\beta(0,1)$-trees and bicubic maps;
- $\beta(1,1)$-trees and 3-connected cubic maps;
- $\beta(2,2)$-trees and cubic non-separable maps; and
- $\beta(1,0)$-trees and non-separable maps.

Also, it is straightforward to see that $\beta(0,0)$-maps are in one-to-one correspondence with rooted plane trees, since one can erase the labels in this case as all of them are 0.

The main results in this paper. One can ask the following question: Is it possible to Gray code a given class of maps? To our best knowledge, no results are known in this direction possibly due to a rather complicated nature of (planar) maps. Thus, one needs to encode the class of maps by words, and then to try to list these words using specified criteria on the number of positions in which the words can differ. Our approach is in encoding the β-description trees in question, which are in a bijective correspondence with the maps of interest, on n vertices by tuples of length $3n-2$; the first $2n-2$ elements of the tuple encode the shape of a

tree (using so-called *Dyck words*), and the remaining elements are used to encode its labels. In either case, for convenience of presentation, we will consider Gray coding shapes of trees separately, which will be given by a known result, while a real challenge will be in (cyclicly) Gray coding β-description trees having the same shape.

We note that β-description trees have already been used to obtain non-trivial equidistribution results on planar maps, e.g. bicubic maps [1], and these trees are a key object in this paper. We will present our results on $\beta(0,1)$-trees, which will give a Gray code for bicubic maps, and then discuss a straightforward extension of that to $\beta(a,b)$-trees with $b \geq 1$. The latter will give at once Gray codes for cubic non-separable maps and 3-connected cubic maps. Thus, our focus will be on $\beta(0,1)$-trees. In particular, the only bijective correspondence we will explain in this paper is that between $\beta(0,1)$-trees and bicubic maps, to give an idea on how bijections between maps and β-description trees corresponding to them could look like; we refer to [2, 3] for bijections between $\beta(1,1)$-trees (resp., $\beta(2,2)$-trees) and 3-connected cubic maps (resp., cubic non-separable maps).

This paper is organized as follows. In Section 2 we discuss $\beta(0,1)$-trees and bicubic maps, in particular sketching a bijection between these sets of objects. In Section 3.1 we discuss a key component in this paper, namely, Gray coding $\beta(0,1)$-trees having the same shape. Cyclic Gray coding $\beta(0,1)$-trees having the same shape is discussed in Section 3.2. Even though Gray coding cyclicly is what we are actually interested in, we first present a Gray code for $\beta(0,1)$-trees having the same shape without the cyclic requirement to prepare the reader for the more involved arguments in the cyclic case. The main results are presented in Section 3.3 along with a definition of Dyck words and necessary results about them. Finally, in Section 4 we provide several directions for further research.

2. $\beta(0,1)$-TREES AND BICUBIC MAPS

Letting $a = 0$ and $b = 1$ in Definition 1 we will obtain a definition of a $\beta(0,1)$-tree. Note that the label of the root of a $\beta(0,1)$-tree is defined uniquely from the labels of its children, which allows us to modify this definition suit better to our purposes. In this paper, we will consider two modifications of the definition. First, we will re-define the root label to be one more than the sum of its children (as was done in [1] for a better description of statistics preserved under the bijection with bicubic maps to be described below), and then we will let the root label be $*$ (to allow two $\beta(0,1)$-trees having the same shape to differ just in one label). Thus, no matter which definition we use, we still have a class of trees in one-to-one correspondence with the originally defined $\beta(0,1)$-trees, and slightly abusing the notation, which will not cause any confusion, we will refer to all of the "modified $\beta(0,1)$-trees" as $\beta(0,1)$-trees.

We continue with stating a slightly modified definition of $\beta(0,1)$-trees, which are particular instances of $\beta(a,b)$-trees introduced in Definition 1.

Definition 2. A $\beta(0,1)$-tree is a rooted plane tree whose vertices are labeled with nonnegative integers such that

  (1) leaves have label 0;
  (2) the label of the root is one more than the sum of its children's labels;
 (3) the label of any other vertex exceeds the sum of its children's labels by at most 1.

The unique $\beta(0,1)$-tree with exactly one vertex (and no edges) is called *trivial*; the root of the trivial tree has label 0. Any other $\beta(0,1)$-tree is called *nontrivial*. In Figure 1, appearing in [1], we have listed all $\beta(0,1)$-trees on 4 vertices. Let
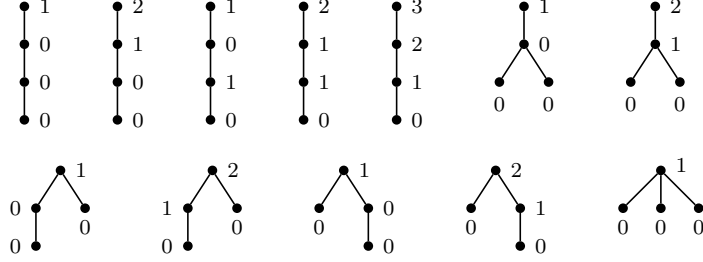
FIGURE 1. All $\beta(0,1)$-trees on 4 vertices.

root(T) denote the root label of T, and let sub(T) denote the number of children of the root. We say that a $\beta(0,1)$-tree T is *reducible* if sub$(T) > 1$, and *irreducible* otherwise. Any reducible tree can be written as a sum of irreducible ones, where the sum $U \oplus V$ of two trees U and V is defined as the tree obtained by identifying the roots of U and V into a new root with label root$(U)+$root$(V)-1$. See Figure 2, taken from [1], for an example.
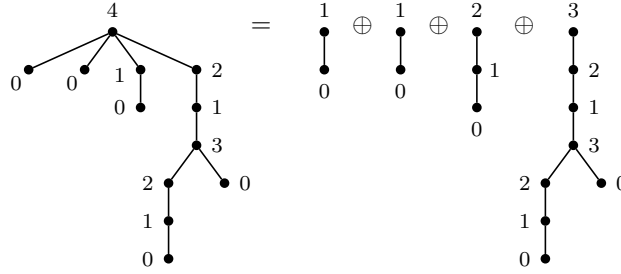
FIGURE 2. Decomposing a reducible $\beta(0,1)$-tree.

Note also that any irreducible tree with at least one edge is of the form $\lambda_i(T)$, where $0 \leq i \leq$ root(T) and $\lambda_i(T)$ is obtained from T by joining a new root via an edge to the old root; the old root is given the label i, and the new root is given the label $i + 1$. For instance,
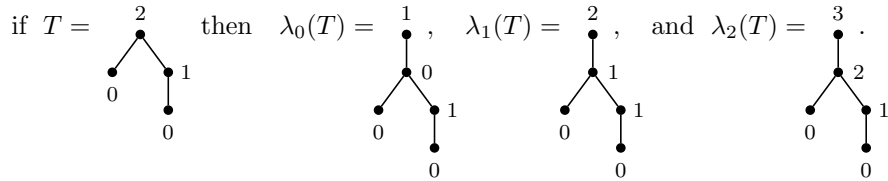
The smallest bicubic map has two vertices and three edges joining them. It is known [3] that the faces of a bicubic map can be colored using three colors so that adjacent faces have distinct colors, say, colors 1, 2 and 3, in a counterclockwise order around white vertices. We will assume that the root vertex is black and the root face has color 3. There are exactly three different bicubic maps with 6 edges and they are given in Figure 3 appearing in [1].

Following [1] we will now describe a bijection between bicubic maps and $\beta(0,1)$-trees. For any bicubic map M and $i = 1, 2, 3$, let $\mathcal{F}_i(M)$ be the set of i-colored faces of M. Let $R_1 \in \mathcal{F}_1(M)$, $R_2 \in \mathcal{F}_2(M)$, and $R_3 \in \mathcal{F}_3(M)$ be the three faces around the root vertex; in particular, R_3 is the root face. In addition, let $S_1 \in \mathcal{F}_1(M)$ be
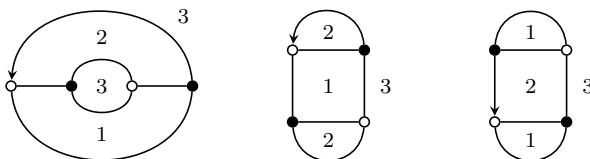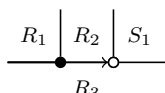
FIGURE 3. All bicubic maps with 4 vertices.

the 1-colored face that meets the vertex that the root edge points to:

We say that a face touches another face k times if there are k different edges each belonging to the boundaries of both faces. Define the following two statistics:

$$\mathtt{f1r3}(M) \text{ is the number of faces in } \mathcal{F}_1(M) \text{ that touch } R_3;$$

$$\mathtt{s1r3}(M) \text{ is the number of times } S_1 \text{ touches } R_3.$$

We say that M is *irreducible* if $\mathtt{s1r3}(M) = 1$, or, in other words, if S_1 touches R_3 exactly once; we say that M is *reducible* otherwise. We shall introduce operations on bicubic maps that correspond to λ_i and $\oplus$ of $\beta(0,1)$-trees. This will induce the desired bijection ψ between bicubic maps and $\beta(0,1)$-trees.

To construct an irreducible bicubic map based on M, and having two more vertices than M, we proceed in one of two ways. The first way (1) corresponds to $\lambda_i(T)$ when $i = \text{root}(T)$; the second way (2) corresponds to $\lambda_i(T)$ when $0 \le i < \text{root}(T)$.

(1) We create a new 1-colored face touching the root face exactly once, so $\mathtt{f1r3}(M') = \mathtt{f1r3}(M) + 1$, by removing the root edge from M and adding a digon that we connect to the map as in Figure 4.
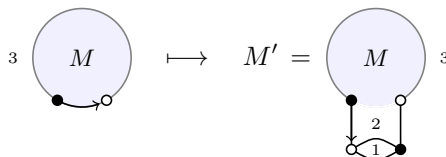

FIGURE 4. Creating an irreducible map.

(2) Assuming that $\mathtt{f1r3}(M) = k$; that is, M has k (different) 1-colored faces touching the root face, we can create an irreducible map M' such that $\mathtt{f1r3}(M') = i$, where $1 \le i \le k$. To this end, we remove the root edge from M. Starting at the root vertex and counting in *clockwise direction*, we also remove the first edge of the ith 1-colored face that touches the root face. In the picture below we schematically illustrate the case $i = 3$. Next we add two more vertices and respective edges, and assign a new root as shown in Figure 5.

Any irreducible bicubic map on $n+2$ vertices can be constructed from some bicubic map on n vertices by applying operation (1) or (2) above.

We shall now describe how to create a reducible map based on irreducible maps M_1, M_2, ..., M_k. An illustration for $k = 3$ can be found in Figure 6. This corresponds to the $\oplus$-operation on $\beta(0,1)$-trees.
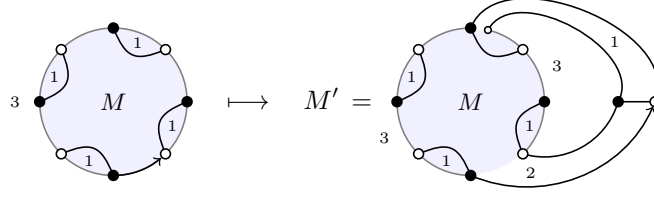
FIGURE 5. The other way to create an irreducible map.

(3) We begin by lining up the maps M_1, M_2, ..., M_k. Next, in each map M_i, we remove the first edge (in *counter-clockwise direction*) from the root edge on the root face. Then we connect the maps as shown in Figure 6, and define the root edge of the obtained map to be the root edge of M_k.
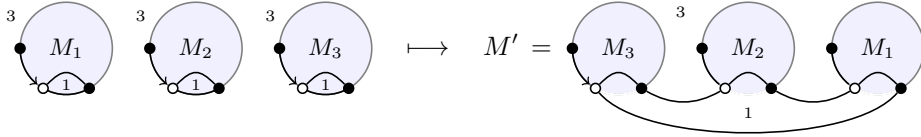

FIGURE 6. Creating a reducible map.

Any reducible bicubic map on n vertices can be constructed by applying the above operation (3) to some ordered list of irreducible bicubic maps whose total number of vertices is n.

By defining operations on bicubic maps corresponding to the operations λ_i and $\oplus$ we have now completed the definition of the bijection ψ between bicubic maps and $\beta(0,1)$-trees. See [1] for examples of non-trivial applications of the bijection.

3. Gray codes

In this section, after introducing some notations, we will define a Gray code for $\beta(0,1)$-trees with the same underlying tree and extend it to a cyclic Gray code for $\beta(0,1)$-trees with arbitrary underlying trees. This will induce a cyclic Gray code for bicubic maps. Then we will see that our construction can be easily extended to a wider class of β-description trees inducing cyclic Gray codes for the corresponding planar maps.

A list $\mathcal{L}$ for a set of length n tuples is a *Gray code* if $\mathcal{L}$ lists, with no repetitions nor omissions, the tuples in the set so that the Hamming distance between two successive tuples in $\mathcal{L}$ (i.e., the number of positions in which they differ) is bounded by a constant, independent of n. And when we want to explicitly refer to this constant, say k, we call such a list a *k-Gray code*. In addition, if the last and first tuple in $\mathcal{L}$ differ in the same way, then the Gray code is *cyclic*.

If $\mathcal{L}$ is a list, then $\overline{\mathcal{L}}$ is the list obtained by reversing $\mathcal{L}$, and if $\mathcal{M}$ is another list, then $\mathcal{L} \circ \mathcal{M}$ is the concatenation of the two lists. If α is a tuple, then $\alpha \cdot \mathcal{L}$ (resp., $\mathcal{L} \cdot \alpha$) is the list obtained by appending (resp., postpending) α to each tuple of $\mathcal{L}$, and α^k is the tuple obtained by concatenating k copies of α. Often we refer to a list by enumerating its elements, e.g. $\mathcal{L} = \langle \alpha_1, \alpha_2, \alpha_3, \ldots \rangle$.

Given a family $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_m\}$ of m lists, each $\mathcal{L}_i$, $1 \leq i \leq m$, being a k_i-Gray code for a set L_i of same length tuples, we define another family of lists $\{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_m\}$ as follows: $\mathcal{N}_1$ is simply the list $\mathcal{L}_m$, and for $2 \leq i \leq m$,

$$\mathcal{N}_i = e_1 \cdot \mathcal{N}_{i-1} \circ e_2 \cdot \overline{\mathcal{N}_{i-1}} \circ e_3 \cdot \mathcal{N}_{i-1} \ldots,$$

where $\langle e_1, e_2, e_3, \ldots, e_j \rangle$ is the list $\mathcal{L}_{m-i+1}$, and the last term of the concatenation defining $\mathcal{N}_i$ is either $e_j \cdot \mathcal{N}_{i-1}$ or $e_j \cdot \overline{\mathcal{N}_{i-1}}$, depending on j being odd or even.

It is routine to prove the following proposition that we will use later.

Proposition 1. *With the notations above, $\mathcal{N}_m$ is a k-Gray code for the product set $L_1 \times L_2 \times \cdots \times L_m$, where $k = \max\{k_1, k_2, \ldots, k_m\}$.*

3.1. **Gray coding $\beta(0,1)$-trees with the same underlying tree.** Recall that by definition, the label of the root of a $\beta(0,1)$-tree is uniquely determined by the labels of its children. In what follows, for convenience, we assume that the root of any $\beta(0,1)$-tree is labeled by $*$. That is, we amend (2) in Definition 2, obtaining a class of trees, still called $\beta(0,1)$-trees by us, which are in one-to-one correspondence with $\beta(0,1)$-trees defined either in Definition 1 or Definition 2. The rationale for (again!) updating slightly our previous definitions is in allowing $\beta(0,1)$-trees to be of distance 1 from each other in the sense specified in Definition 6 below.

We are also interested in the following class of labeled trees, which are essentially $\beta(0,1)$-trees, but where the root is treated as any other internal vertex.

Definition 3. A $\beta'(0,1)$-tree is a rooted plane tree whose vertices are labeled with nonnegative integers such that

    (1) leaves have label 0;
    (2) the label of any other vertex exceeds the sum of its children's labels by at most 1.

Note that any $\beta(0,1)$-tree discussed in Definition 2 is a $\beta'(0,1)$-tree.

Definition 4. Let T be a $\beta(0,1)$-tree or a $\beta'(0,1)$-tree. Then $u(T)$ denotes the underlying rooted tree, that is, the tree obtained by removing all labels in T. In other words, $u(T)$ gives the shape of T.

For example, if T is the rightmost $\beta(0,1)$-tree in the top row in Figure 1, then $u(T)$ is given by

Definition 5. Let T be a $\beta(0,1)$-tree or a $\beta'(0,1)$-tree with n vertices. We let $\ell(T)$ denote the n-tuple of T's labels obtained by traversing T by depth first search using the leftmost option and reading each label exactly once.

For example, for the tree T in Figure 2, $\ell(T) = (4, 0, 0, 1, 0, 2, 1, 3, 2, 1, 0, 0)$; see also Table 2 where the roots are labeled by *. Thus, $\ell(T)$ is an encoding of a given $\beta(0,1)$-tree T in the form of a tuple. Note that this encoding disregards the shape of the tree.

Definition 6. For $\beta(0,1)$-trees (with the root labeled by *), or $\beta'(0,1)$-trees, T_1 and T_2 such that $u(T_1) = u(T_2)$, the *distance* $d(T_1, T_2)$ between the trees is the number of positions in which $\ell(T_1)$ and $\ell(T_2)$ differ (or equivalently, the Hamming distance between $\ell(T_1)$ and $\ell(T_2)$).

For example, keeping in mind that we label the root of a $\beta(0,1)$-tree by $*$, the distance between the first and the fourth trees in the top row in Figure 1 is 2, while the distance between the first and the second trees in the bottom row in that figure is 1.

Definition 7. For a $\beta'(0,1)$-tree T, we let $L(T)$ denote the set of encodings of all $\beta'(0,1)$-trees obtained by labeling properly $u(T)$.

Lemma 2. *For any $\beta'(0,1)$-tree T there is a 1-Gray code for $L(T)$.*

Before giving a formal proof of Lemma 2, we explain its general idea, which is presented graphically in Figure 7(a). Assuming the existence of a 1-Gray code for smaller trees, we can extend such a code to trees obtained by adding the new root. More precisely, each vertex in Figure 7(a) corresponds to a $\beta'(0,1)$-tree having a fixed shape (that is, underlying tree). Vertices on the same vertical line correspond to $\beta'(0,1)$-trees that differ only in the root label: the higher a vertex is, the larger root label it corresponds to. Note that each vertical line must contain at least the vertices corresponding to root labels 0 and 1, but it may or may not contain other vertices.

Further, a_1, a_2, etc. in this figure form a Hamiltonian path corresponding to the 1-Gray code for the trees with root label equal to 0, whose existence we assumed. Also, b_1, b_2, etc. is such a path for the trees with root label equal to 1. Thus, the trees corresponding to a_i and b_i differ only in the root label. The desired Hamiltonian path through all the $\beta'(0,1)$-trees (corresponding to the 1-Gray code for $L(T)$) presented schematically in Figure 7(a) begins at a_1 and goes in the direction of the arrow.

Proof of Lemma 2. We proceed by induction on the number v of vertices in T. The base cases, $v=1$ (the single vertex $\beta'(0,1)$-tree) and $v=2$ (two single edge $\beta'(0,1)$-trees, which are on distance 1 from each other) obviously hold.

Suppose now that $v \geq 3$, and the children of the root are the roots of subtrees $T_1, T_2, \ldots, T_k$ from left to right, where $k \geq 1$. Each T_i is a $\beta'(0,1)$-tree and by induction hypothesis, $L(T_i)$ has a 1-Gray code. But then, by Proposition 1,

$$L(T_1) \times L(T_2) \times \cdots \times L(T_k)$$

also has a 1-Gray code, and it can be extended to a 1-Gray code of $L(T)$ obtained by adding the new leftmost coordinate corresponding to T's root to each entry of $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ as explained below.

For an integer $u \geq 0$ we define two lists of 1-tuples:

- $\gamma(u) = \langle (0), (u), (u-1), \ldots, (2), (1) \rangle$, and
- $\delta(u) = \langle (1), (u), (u-1), \ldots, (2), (0) \rangle$.

In particular, $\gamma(1)$ and $\delta(1)$ are the lists $\langle (0), (1) \rangle$ and $\langle (1), (0) \rangle$, respectively.

Let $\langle \alpha_1, \alpha_2, \ldots \rangle$ be the 1-Gray code list for $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$, so that each α_j is the concatenation of k tuples corresponding to the labels of the vertices of the trees $T_1, T_2, \ldots, T_k$, and let $m(\alpha_j)$ be the sum of the labels of the roots of these trees plus one. In other words, $m(\alpha_j)$ is the maximal value of x, such that $(x) \cdot \alpha_j$ is a proper labeling of T. Thus $m(\alpha_j) \geq 1$ and $m(\alpha_j) = 1$ if and only if the root of each T_i is labeled by 0. Finally, let $\mathcal{M}$ be the list defined as

$$\mathcal{M} = \mathcal{M}_1 \circ \mathcal{M}_2 \circ \mathcal{M}_3 \circ \cdots$$

with

$$\mathcal{M}_j = \begin{cases} \gamma(m(\alpha_j)) \cdot \alpha_j & \text{if } j \text{ is odd,} \\ \delta(m(\alpha_j)) \cdot \alpha_j & \text{if } j \text{ is even.} \end{cases}$$

(a) (b)

FIGURE 7. Schematic approach in the proof of (a) Lemma 2, and
(b) Lemma 6.

Clearly, the underlying set of $\mathcal{M}$ is $L(T)$. In addition $\mathcal{M}$ is a 1-Gray code: throughout each list $\mathcal{M}_j$ successive tuples differ in the first position, and the last element of $\mathcal{M}_j$ differ from the first element of $\mathcal{M}_{j+1}$ as α_j differ from α_{j+1}, that is in a single position.

Thus, $L(T)$ has a 1-Gray code and the statement is proved by induction. $\qquad\square$

Theorem 3. *There exists a* 1*-Gray code for* $\beta(0,1)$*-trees having the same underlying tree.*

Proof. Suppose that the root of a $\beta(0,1)$-tree T, labeled by $*$, has subtrees $T_1, \ldots, T_k$, where $k \geq 1$. Each T_i is a $\beta'(0,1)$-tree, and thus, by Lemma 2, there is a 1-Gray code for each $L(T_i)$. But then, by Proposition 1, there is also a 1-Gray code for $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ leading to the fact that $L(T) = \{(*)\} \times L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ has a 1-Gray code, as desired. $\qquad\square$

Note that generally speaking the 1-Gray codes in Lemma 2 and Theorem 3 are not cyclic, and below we discuss how cyclic 1-Gray codes in this context can be obtained.

3.2. **Gray coding cyclicly $\beta(0,1)$-trees with the same underlying tree.** The results presented in the previous subsection can be generalized to cyclic Gray codes in question. The goal of this subsection is to justify this, and in contrast with the previous subsection, here the proofs will be rather existential than constructive. First note that Proposition 1 can be generalized to the following proposition that is easy to prove directly, but also it follows from more general results presented in [4].

Proposition 4. *Suppose that $L_1, L_2, \ldots, L_m$ are sets, and each of them is a set of same length tuples which is either a singleton, or there is a cyclic* 1*-Gray code for it. Then there is a cyclic* 1*-Gray code for the product set $L_1 \times L_2 \times \cdots \times L_m$.*

In what follows we will need the following easy to understand facts.

**Fact 5.** *Let $\mathcal{G}$ be a* 1*-Gray code for a set of length k tuples.*
- *If $r = (r_1, \ldots, r_k)$ and $s = (s_1, \ldots, s_k)$ are consecutive tuples in $\mathcal{G}$, then there are no i and j, $1 \leq i \neq j \leq k$, such that $r_i \neq s_i$ and $r_j \neq s_j$.*
- *If $t = (t_1, \ldots, t_k)$ is a third tuple, and r, s and t are consecutive (in this order) in $\mathcal{G}$ and there is an i with $s_i \notin \{r_i, t_i\}$, then $r_i \neq t_i$.*

As mentioned before, the Gray code defined in Lemma 2 is not necessarily cyclic. However, if the number of a_is in the construction of the Hamiltonian path P in Figure 7(a) is even, then the assumption of existence of a Hamiltonian cycle (or equivalently, of a cyclic 1-Gray code) for smaller $\beta'(0,1)$-trees would give us the cyclicity of the path P, and so, the cyclicity of the Gray code defined in Lemma 2.

Indeed, in this case the first and the last vertices in P are a_1 and a_m, where m is the maximum index, and a_1 and a_m are connected by an edge.

Thus, the difficult case is when the number of a_is is odd, which is possible (e.g. there are five $\beta'(0,1)$-trees having the shape of a path on three vertices). To handle this situation, we use the idea presented in Figure 7(b). Namely, we will prove that essentially in all cases, there is an edge $c_1 c_2$ between two $\beta'(0,1)$-trees having root label 2 such that the edges corresponding to it on levels "root label = 0" and "root label = 1" are involved in the respective Hamiltonian cycles assumed by the induction hypothesis (these edges are $a_k a_{k+1}$ and $b_k b_{k+1}$ in Figure 7(b)). The existence of $c_1 c_2$ allows us to change the turns taken from the two Hamiltonian cycles making sure that the first and the last vertices in P will be a_1 and a_m. In what follows, for a given tree we will refer to a vertex different from the root or a leaf as *internal vertex*. The two situations when $c_1 c_2$ does not exist are easy to handle. This happens when

- there are no internal vertices in a tree. For a given number of vertices, there are only two such $\beta'(0,1)$-trees, any listing of which gives a cyclic 1-Gray code;
- there is exactly one internal vertex in a tree. One can easily check that, for any tree T with exactly one internal vertex, there are five $\beta'(0,1)$-trees with the shape of T, and the number of such trees with root label 0 (equivalently, 1), which is the number of a_i's (equivalently, b_i's) is two, which is even, so that the existence of a Hamiltonian cycle in this case is easy to establish using the approach in Figure 7(a). For example, the (ordered) list below is a cyclic 1-Gray code for the $\beta'(0,1)$-trees of same shape on four vertices:
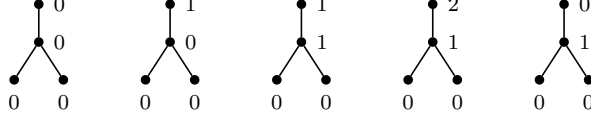
Recall from Definition 7 that for a $\beta'(0,1)$-tree T, $L(T)$ denotes the set of encodings of all $\beta'(0,1)$-trees obtained by labeling properly $u(T)$. The following lemma generalizes Lemma 2.

Lemma 6. *For any $\beta'(0,1)$-tree T there is a cyclic 1-Gray code for $L(T)$.*

Proof. Based on the discussion preceding the statement of this lemma, the proof of Lemma 2 can be used if we will prove the existence of an edge $c_1 c_2$ in the situation when T has at least two internal vertices. If the subtrees of T are $T_1, T_2 \ldots, T_k$, the existence of $c_1 c_2$ is equivalent with the existence of two successive tuples in the Gray code for the product set $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ which both can be extended to proper labeling of T by letting the root of T be 2. To this end, we consider two subcases.

- The root of T has exactly one internal vertex v among its children. It follows that, T_j, the subtree rooted in v in turn has at least one internal vertex among its children. It is easily seen that the product set $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ collapses to $(0, \ldots, 0) \cdot L(T_j) \cdot (0, \ldots, 0)$, for appropriate length 0s prefix and suffix, and a Gray code for this product set is essentially a Gray code for $L(T_j)$. Let u be a tuple in $L(T_j)$ representing a proper labeling of T_j where the label of v (that is, the first entry of u) is a non-zero value. Since T_j has at least one internal vertex it follows that there are at least three such u, and we choose one which is not the last nor the first tuple in the Gray code for $L(T_j)$. The existence of this Gray code follows by inductive hypothesis. By the second point of Fact 5, at least one among

the successor and the predecessor of u, in the Gray code for $L(T_j)$, has a non-zero value in the position corresponding to v, the root of T_j. It follows that there are two successive tuples in the Gray code for $L(T_j)$, and so for the above product set, which can be extended to a proper labeling of T by letting the root of T be 2, which is allowed by the rules of $\beta'(0,1)$-trees; and this extension will give us the desired edge $c_1 c_2$.

- The root of T has at least two internal vertices among its children. Let i and j be the positions corresponding to the labels of two of these vertices in the tuples in $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$. Clearly, 1 is an admissible value for the entries in positions i and j. Let u be a tuple in $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ having 1 in both positions i and j. By the first point of Fact 5 it follows that if u is not the last tuple in the Gray code for the product set $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ (whose existence we assume by inductive hypothesis and Proposition 4), then the successor of u has 1 in at least one of these two positions. The reasoning is similar when u is the last tuple by replacing "successor" by "predecessor". And again, it follows that there are two successive tuples in the Gray code for $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ which can be extended to proper labeling of T by letting the label of the root of T be 2.

Thus, in both cases there is a 1-Gray code for $L(T)$ with the first and last tuple of the form $(0, \alpha_1, \alpha_2, \dots)$ and $(0, \omega_1, \omega_2, \dots)$, respectively, where $(\alpha_1, \alpha_2, \dots)$ and $(\omega_1, \omega_2, \dots)$ are the first and last tuples in the cyclic Gray code for the product set $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$, and the statement follows. $\qquad\square$

Using Lemma 6, we can now generalize Theorem 3, which is the main result in this subsection.

Theorem 7. *There exists a cyclic 1-Gray code for $\beta(0,1)$-trees having the same underlying tree.*

Proof. Suppose that the root of a $\beta(0,1)$-tree T, labeled by $*$, has subtrees $T_1, \dots, T_k$, where $k \geq 1$. Each T_i is a $\beta'(0,1)$-tree, and thus, by Lemma 6, there is a cyclic 1-Gray code for each $L(T_i)$. But then, by Proposition 4, there is also a cyclic 1-Gray code for $L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ leading to the fact that $L(T) = \{(*)\} \times L(T_1) \times L(T_2) \times \cdots \times L(T_k)$ has a cyclic 1-Gray code, as desired. $\quad\square$

Since the tuple $(*, 0, 0, \dots, 0)$ of appropriate length is always an admissible encoding of a $\beta(0,1)$-trees, we have:

Corollary 8. *There exists a 1-Gray code list for $\beta(0,1)$-trees having the same underlying tree which begins by $(*, 0, 0, \dots, 0)$ and ends by a tuple differing from $(*, 0, 0, \dots, 0)$ in exactly one position.*

3.3. **Dyck words and Gray coding bicubic maps.** For two integers k and m, $0 \leq k \leq m$, we denote by $D_{m,k}$ the set of binary tuples with m occurrences of 1 and k occurrences of 0, satisfying the *prefix property*: no prefix contains more 0s than 1s. For example, $(1, 0, 1, 1, 0, 1, 1, 0, 0) \in D_{5,4}$. The set $D_{m,m}$ is known as the set of *Dyck words* of length $2m$. The number of elements in $D_{m,m}$ is the well known mth *Catalan number* $C_m = \frac{1}{m+1}\binom{2m}{m}$. See Table 1 for the 14 length 8 Dyck words.

Traversing a plane tree by depth first search using the leftmost option and letting 1 represent a forward step and 0 represent a backward step, we obtain a one-to-one correspondence between plane trees on $n+1$ vertices (and n edges) and Dyck words of length $2n$. We let $w(T)$ denote the Dyck word corresponding to a plane tree T under this well known and easy to prove bijection. For example, for the tree T on 4 vertices illustrating Definition 4, $w(T) = (1, 1, 0, 1, 0, 0)$; see also Figure 2.

$$
\begin{array}{|c|}
\hline
(1, 0, 1, 0, 1, 0, 1, 0) \\
(1, 1, 0, 0, 1, 0, 1, 0) \\
(1, 1, 1, 0, 0, 0, 1, 0) \\
(1, 1, 0, 1, 0, 0, 1, 0) \\
(1, 0, 1, 1, 0, 0, 1, 0) \\
(1, 0, 1, 1, 1, 0, 0, 0) \\
(1, 1, 0, 1, 1, 0, 0, 0) \\
(1, 1, 1, 0, 1, 0, 0, 0) \\
(1, 1, 1, 1, 0, 0, 0, 0) \\
(1, 0, 1, 1, 0, 1, 0, 0) \\
(1, 1, 0, 1, 0, 1, 0, 0) \\
(1, 1, 1, 0, 0, 1, 0, 0) \\
(1, 1, 0, 0, 1, 1, 0, 0) \\
(1, 0, 1, 0, 1, 1, 0, 0) \\
\hline
\end{array}
$$

TABLE 1. The Gray code list $\mathcal{D}_{4,4}$ for the set of length 8 Dyck words.

Note that the Hamming distance between two same length Dyck words is always even, and thus the minimum distance between two Dyck words is 2.

The following recursive description obtained in [11] gives a Gray code for $D_{m,k}$, and, in particular, for the set of length $2m$ Dyck words; this description is a slight variation of the code defined in [8]:

$$
\mathcal{D}_{m,k} = \begin{cases}
(1)^m & \text{if} \quad k = 0, \\
\mathcal{D}_{m,k-1} \cdot (0) & \text{if} \quad m = k > 0, \\
\mathcal{D}_{m-1,k} \cdot (1) \circ \overline{\mathcal{D}}_{m,k-1} \cdot (0) & \text{if} \quad m > k > 0.
\end{cases}
\tag{1}
$$

See Table 1, showing $\mathcal{D}_{4,4}$, for an example. The following lemma was proved in [11].

Lemma 9. [11] *The list $\mathcal{D}_{m,k}$ satisfies the following properties:*

- *The first tuple in $\mathcal{D}_{m,k}$ is $(1,0)^k \cdot (1)^{m-k}$;*
- *The last tuple in $\mathcal{D}_{m,k}$ is*
  - *$(1,0)^{m-2} \cdot (1,1,0,0)$, if $k = m > 1$,*
  - *$(1,0)^{k-1} \cdot (1)^{m-k+1} \cdot (0)$, if $m > k \geq 1$ or $m = k = 1$,*
  - *$(1)^m$, if $m > k = 0$;*
- *Two successive tuples in $\mathcal{D}_{m,k}$, including the last and the first one, differ in exactly two positions, and thus $\mathcal{D}_{m,k}$ is a cyclic 2-Gray code.*

In what follows, the Hamming distance between tuples is denoted by d, and the next definition extends it to trees.

Definition 8. For $\beta(0,1)$-trees T_1 and T_2 on the same number of vertices, the *distance $d(T_1, T_2)$* between the trees is defined as

$$
d(T_1, T_2) = d(\ell(T_1), \ell(T_2)) + d(w(u(T_1)), w(u(T_2))).
$$

Theorem 10. *There exists a cyclic 3-Gray code for $\beta(0,1)$-trees (with the root labeled by *) on n vertices, $n \geq 1$, with respect to the distance given in Definition 8.*

Proof. Any $\beta(0,1)$-tree T on n vertices can be encoded by a $(3n-2)$-tuple, which is obtained by merging the $(2n-2)$-tuples $w(u(T))$ and the n-tuples $\ell(T)$. Let

$$
d_1 \cdot \mathcal{L}(T(d_1)) \circ d_2 \cdot \mathcal{L}(T(d_2)) \circ d_3 \cdot \mathcal{L}(T(d_3)) \circ \cdots
\tag{2}
$$

be the list where d_i is the ith tuple in the list $\mathcal{D}_{n-1,n-1}$ defined in relation (1), $T(d_i)$ is the tree encoded by d_i, and $\mathcal{L}(T(d_i))$ the list assumed by Corollary 8 for the $\beta(0,1)$-trees having the shape $T(d_i)$.

It is easy to see that in the list defined in relation (2) two successive tuples are at distance at most 3. Indeed,

- for a fixed d_i, successive tuples in $d_i \cdot \mathcal{L}(T(d_i))$ differ in one position, and
- for two successive tuples d_i and d_{i+1} in $\mathcal{D}_{n-1,n-1}$ (including the last and the first ones), the last tuple in $d_i \cdot \mathcal{L}(T(d_i))$ and the first one in $d_{i+1} \cdot \mathcal{L}(T(d_{i+1}))$ differ in three positions.

$\square$

Note that the Gray code stated in Theorem 10 for $\beta(0,1)$-trees is minimal, in the sense that, in general there are no cyclic 2-Gray codes for $\beta(0,1)$-trees. See Table 2 for an example where $\beta(0,1)$-trees, encoded by (1,1,0,0,*,0,0), (1,1,0,0,*,1,0) and (1,0,1,0,*,0,0), cannot be listed cyclically so that the distance between successive trees is at most 2. Also, the Gray code defined in (2) is "shape partitioned", that is, same shape $\beta(0,1)$-trees are successive in it.

| T			
$\ell(T)$	(*,0,0)	(*,1,0)	(*,0,0)
$w(u(T))$	(1,1,0,0)	(1,1,0,0)	(1,0,1,0)

TABLE 2. The three $\beta(0,1)$-trees on three vertices with the root labeled by *, and the corresponding to them the depth first leftmost option reading of the labels, and the Dyck words coding their shape.

Our way to Gray code bicubic maps can be applied to any class of planar maps that can be described in terms of $\beta(a,b)$-trees with $b \geq 1$. Namely, generalizing the notion of $\beta'(0,1)$-trees to that of $\beta'(a,b)$-trees (by removing the condition on the root in Definition 1), we can essentially copy/paste all our arguments for $\beta'(0,1)$-trees. Indeed, for such a $\beta'(a,b)$-tree T, the levels corresponding to the root's label a and $a+1$ will be isomorphic, so that induction can be used in the way we used it for $\beta'(0,1)$-trees. Thus, in particular, we can Gray code 3-connected cubic planar maps and cubic non-separable planar maps corresponding to $\beta(1,1)$-trees and $\beta(2,2)$-trees, respectively [2, 3].

Finally, note that having $b \geq 2$ would simplify some of our arguments. In particular, in this case there is no need to prove the existence of the edge $c_1 c_2$ in Lemma 6, since we will have at least three isomorphic levels of vertices corresponding to the root labels a, $a+1$ and $a+2$, so that existence of an edge with the right properties will be given to us automatically (in fact, each edge from the Hamiltonian path on level $a+2$ will have the right properties).

4. Concluding remarks

In this paper we have shown that classes of planar maps corresponding to $\beta(a,b)$-trees with $b \geq 1$ have cyclic 3-Gray codes, and these codes are minimal in the sense of Hamming distance. We leave it as an open problem to determine whether there exist (cyclic) k-Gray codes, for some $k \geq 1$, for $\beta(a,0)$-trees, where $a \geq 1$. In the case $a = 1$ such a code would induce a Gray code on non-separable planar maps via the respective bijection [2].

It would also be interesting to Gray code so-called α-description trees (see [2] for the definition) that would induce Gray coding of *planar maps* and *Eulerian planar maps* [2].

Acknowledgments

References

[1] A. Claesson, S. Kitaev and A. de Mier. An involution on bicubic maps and beta(0,1)-trees, *Australasian J. Combin.* **61(1)** (2015) 1–18.

[2] R. Cori, B. Jacquard and G. Schaeffer. Description trees for some families of planar maps, *Formal Power Series and Algebraic Combinatorics* (1997) 196–208. Proceedings of the 9th Conference, Vienna.

[3] R. Cori and G. Schaeffer. Description trees and Tutte formulas, *Theoret. Comput. Sci.* **292** (1997), 165–183.

[4] V. V. Dimakopoulos, L. Palios and A. S. Poulakidas. On the Hamiltonicity of the Cartesian product. *Inform. Process. Lett.* **96** (2005), no. 2, 49–53.

[5] R. W. Doran. The Gray code, *J. UCS* **13(11)** (2007) 1573–1597.

[6] 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** (2008) 35–49.

[7] S. Kitaev. Patterns in permutations and words, Springer-Verlag, 2011.

[8] F. Ruskey and A. Proskurowski. Generating binary trees by transpositions, *Journal of Algorithms* **11** (1990) 68–84.

[9] F. Ruskey. Combinatorial gseneration, book in preparation.

[10] C. Savage. A survey of combinatorial Gray codes, *SIAM Rev.* **39(4)** (2006) 605–629.

[11] V. Vajnovszki, Generating a Gray code for P–sequences, *IJMA* **1** (2002) 31–41.

[12] W. T. Tutte. A census of planar maps, *Canad. J. Math.* **67** (1963) 15, 249–271.

[13] W. T. Tutte. *Graph Theory As I Have Known It*, Oxford University Press, New York, 1998.

[14] T. Walsh. Generating Gray Codes in $O(1)$ worst-case time per word, *4*th *Discrete Mathematics and Theoretical Computer Science Conference*, Dijon-France, 7–12 July 2003 (LNCS, 2731, 73–88).

Sobolev Institute of Mathematics, 4 Acad. Koptyug Ave, 630090 Novosibirsk, Russia, avgust@math.nsc.ru

Department of Computer and Information Sciences, University of Strathclyde, 26 Richmond Street, Glasgow G1 1XH, United Kingdom, sergey.kitaev@strath.ac.uk

Sobolev Institute of Mathematics, 4 Acad. Koptyug Ave, 630090 Novosibirsk, Russia, vpotapov@math.nsc.ru

LE2I, Université de Bourgogne Franche-Comté, BP 47870, 21078 Dijon Cedex, France, vvajnov@u-bourgogne.fr