

## Canonical Decomposition of Outerplanar Maps and Application to Enumeration, Coding and Generation

*Nicolas Bonichon Cyril Gavoille Nicolas Hanusse*

CNRS-LaBRI

Université Bordeaux I

{bonichon,gavoille,hanusse}@labri.fr

### Abstract

In this article we define a canonical decomposition of rooted outerplanar maps into a spanning tree and a list of edges. This decomposition, constructible in linear time in the Word-RAM model, implies the existence of bijection between rooted outerplanar maps with  $n$  nodes and bicolored rooted ordered trees with  $n$  nodes where all the nodes of the last branch are colored white. As a consequence, for rooted outerplanar maps of  $n$  nodes, we derive:

- an enumeration formula, and an asymptotic of  $2^{3n-\Theta(\log n)}$ ;
- an optimal data structure of asymptotically  $3n$  bits, built in  $O(n)$  time, supporting adjacency and degree queries in worst-case constant time and neighbors query of a degree- $d$  node in worst-case  $O(d)$  time.
- an  $O(n)$  expected time uniform random generating algorithm.

Article Type	Communicated by	Submitted	Revised
Regular paper	X. He	July 2003	August 2005

---

An extended abstract of this article appeared in Proceeding of WG'2003, Lecture Notes in Computer Science 2880, pp. 81-92. This work was supported by the ACI Masse de données grant of French Ministry of Research under Navgraphe project.

## 1 Introduction

A graph is *outerplanar* if it can be drawn on the plane with non-intersecting edges such that all the nodes lie on the boundary of the infinite face, also called *outerface*. Characterization of outerplanar graphs has been given by Chartrand and Harary [8]: a graph is outerplanar if and only if it has neither  $K_{2,3}$  nor  $K_4$  as a minor. A linear time recognition algorithm has been given by Mitchell [19]. Labeled and unlabeled outerplanar graphs can be randomly generated in  $O(n^4 \log n)$  space and  $O(n^2)$  time [5] after a preprocessing of  $O(n^5)$  time. Among graph properties, outerplanar graphs contain trees, have tree-width at most two, and are exactly the graphs of *pagenumber* one [4]. Recall that a graph  $G$  has pagenumber  $k$  if  $k$  is the smaller integer for which  $G$  has a  $k$ -*page* embedding, also called *book* embedding. In such an embedding the nodes are drawn on a straight line (the spine of a book), and the edges are partitioned into  $k$  *pages*, each page consisting of non-intersecting edges. An outerplanar graph with a 1-page embedding is depicted on Figure 1c).

A *planar map* is a connected graph embedded on the sphere with non-intersecting edges (see [11] for a survey). A planar map is *outerplanar* if all the nodes lie on one face, called the *outerface*. For convenience, outerplanar maps are drawn on the plane such that the *outerface* corresponds to the infinite face, see Figure 1b). A map is *rooted* if one of its edges, the *root*, is distinguished and oriented. In this case, the map is drawn on the plane in such a way that whenever traveling clockwise around the boundary of the *outerface*, the tail of the root edge is traversed before its head. A *planted tree* is the rooted planar map of a rooted tree such that the tail of the root of the map coincides with the root of the tree. In the literature, planted trees are also named ordered rooted trees. All the maps considered in this paper are planar, rooted, and are simple (have no loops and multi-edges).

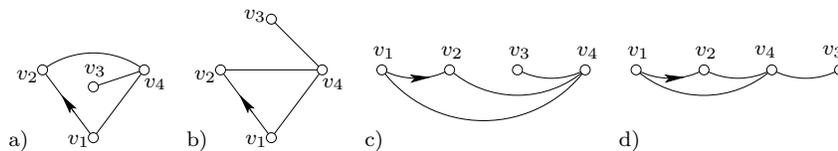


Figure 1: Four drawings for an outerplanar graph  $G$ : a) a planar (but not outerplanar) map; b) an outerplanar map; and c) a 1-page embedding. As graphs, a), b), c) and d) are identical, but as maps, only b), c) and d) are identical. As 1-page embedding, we point out that c) and d) are different.

Some sub-classes of outerplanar maps are well-known. Planted trees and maximal outerplanar maps, i.e., the map of an outerplanar graph with the maximum number of edges, are counted by the Catalan numbers. Finally, bi-connected outerplanar maps can be seen as dissections of a convex polygon, and their number can be counted by Schröder numbers. For these three sub-classes there exist linear time random generation algorithms [1, 3, 14].

Besides the combinatorial aspect and random generation, a lot of attention is given in Computer Science to *efficiently* represent discrete objects. By “efficiently”, we mean that the representation should be succinct, i.e., the storage of these objects requires few bits, and that the time to compute such representation should be polynomial in its size. Fast manipulation of the so encoded objects and easy access to a part of the code are also desirable properties. Typically, adjacency query, i.e., check if two nodes are neighbors or not, and degree query, i.e., how many neighbors a node has, should be given very fast.

For instance, a folklore encoding of  $n$ -edge planted trees, based on a clockwise depth-first traversal, yields a representation with  $2n - O(1)$  bits. This coding length is asymptotically optimal since the number of possible  $n$ -edge planted trees is the  $n$ th Catalan number  $\frac{1}{n+1} \binom{2n}{n} \sim 2^{2n - O(\log n)}$ . Completing this coding by an efficient data structures of length  $o(n)$  bits, it has been shown in [9, 20] that adjacency and degree queries can then be answered in constant time, that the neighbors of a degree- $d$  node can be determined in  $O(d)$  time, assuming that: 1) nodes of the tree are labeled according to the depth-first traversal (i.e., node  $i$  must be the  $i$ th node encountered in the clockwise prefix order of the tree); and 2) standard arithmetic operations on integers of  $O(\log n)$  bits can be performed in constant time.

Outerplanar graphs are an interesting class of graphs because they are isomorphic to graphs ofpagenumber one, cf. Figure 1. Our contribution is an optimal  $3n$ -bit encoding for outerplanar maps. We point out that there exist many 1-page embeddings for a graph ofpagenumber one. From the asymptotic formula of Flajolet and Noy [16], any encoding of 1-page embeddings requires  $3.37n$  bits<sup>1</sup>.

Let us sketch our technique. First we show that an outerplanar map admits a canonical decomposition into a particular rooted spanning tree (called *well-orderly tree* and defined in Section 2), and a set of additional edges  $(u, v)$  such that  $v$  is the first node after  $u$  (in a clockwise preorder of the tree) that is not a descendant of  $v$ . This decomposition can be computed in linear time. Then, we give three applications to this decomposition: enumeration formula (Section 3), efficient encoding (Section 4), and random generation algorithm with experiments (Section 5).

Hereafter we denote by  $T_{n,d}$  the number of  $n$ -node planted trees where the depth of the clockwise last leaf is  $d$  (formulas for these numbers are given in [15, 18]). Our canonical decomposition gives a bijection between outerplanar maps and bicolored trees (more precisely planted trees in which the nodes are colored either white or black), and where all the nodes (including the root) of the clockwise last branch are colored white. Clearly these last objects are counted by the following numbers:

$$\sum_{d=1}^{n-1} 2^{n-d-1} T_{n,d} .$$

From this bijection, we show that the number of  $n$ -node outerplanar maps is

---

<sup>1</sup>In their article, 1-page embeddings are called non-crossing graphs.

asymptotically

$$\frac{2^{3n}}{36n\sqrt{\pi n}}.$$

Moreover, the average number of edges of an outerplanar map is  $\frac{3}{2}n - O(1)$ .

An information-theoretic optimal encoding algorithm is deduced from the previous decomposition. It takes a  $O(n)$  time in the Word-RAM Model and uses at most  $3n - 6$  bits, for  $n \geq 2$ , as follows:  $2n - 4$  bits are used to encode the spanning tree, and  $n - 2$  bits (at most) are used to encode the additional edges. Adding a standard  $o(n)$ -bit data structure to this coding [20, 9], adjacency and degree queries can then be answered in worst-case constant time and neighbors of a  $d$ -degree node can be determined in  $O(d)$ .

Using a grammar to produce bicolored rooted ordered trees with  $n$  nodes where all the nodes of the last branch are colored white, and using Goldwurm's algorithm [17], a random outerplanar map can be generated uniformly with  $O(n)$  space and  $O(n^2)$  average time. Using Floating-Point Arithmetic [12], this average time complexity can be reduced to  $O(n^{1+\epsilon})$ . In Section 5, we propose an  $O(n)$  expected time and  $O(n)$  space complexity generating algorithm. It can generate outerplanar maps with a given number of nodes, or with a given number of nodes and of edges.

## 2 Well-Orderly Tree of an Outerplanar Map

In [7] the authors introduced the *well-orderly trees*, a special case of the *orderly spanning trees* [9]. Let  $T$  be a rooted spanning tree of a planar map  $H$ . Two nodes are *unrelated* if neither of them is an ancestor of the other in  $T$ . An edge of  $H$  is unrelated if its endpoints are unrelated. Let  $v_1, v_2, \dots, v_n$  be the clockwise preordering of the nodes in  $T$ . A node  $v_i$  is *well-orderly* in  $H$  w.r.t.  $T$  if the incident edges of  $v_i$  in  $H$  form the following blocks (possibly empty) in clockwise order around  $v_i$ :

- $B_P(v_i)$ : the edge incident to the parent of  $v_i$ ;
- $B_<(v_i)$ : unrelated edges incident to nodes  $v_j$  with  $j < i$ ;
- $B_C(v_i)$ : edges incident to the children of  $v_i$ ;
- $B_>(v_i)$ : unrelated edges incident to nodes  $v_j$  with  $j > i$ ; and
- the clockwise first edge  $(v_i, v_j) \in B_>(v_i)$ , if it exists, verifies that  $v_i$  is a descendant of the parent of  $v_j$ ;

$T$  is a *well-orderly tree* of  $H$  if all the nodes of  $T$  are well-orderly in  $H$ , and if the root of  $T$  belongs to the boundary of the outerface of  $H$  (see Figure 2 for an example). Note that a well-orderly tree is necessarily a spanning tree. Observe also that for every edge  $e$  of  $H$ , with  $e = (v_i, v_j)$  and  $i < j$ , we have either  $e \in B_C(v_i)$  (i.e.,  $e \in T$ ), or  $e \in B_>(v_i)$ . For convenience, the clockwise first edge of  $B_>(v_i)$ , if it exists, is called the *front edge* of  $v_i$ .

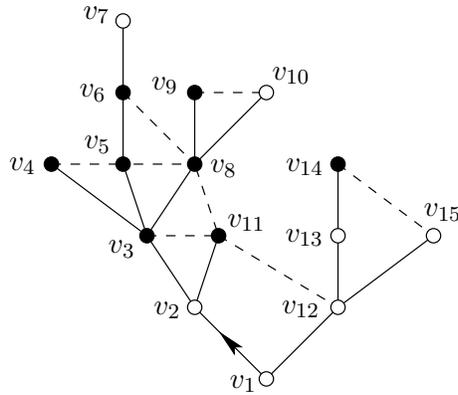


Figure 2: A rooted outerplanar map, and its well-orderly tree depicted with solid edges.

Not all planar maps admit a well-orderly tree. For instance the map of Figure 1a) has no well-orderly tree rooted at the tail of the root edge, whereas the map of Figure 1b) has one. Actually, we have:

**Lemma 1** ([7]) *Every rooted planar map admits at most one well-orderly tree rooted at the tail of the root edge of the map.*

The proof of this lemma can be found in [6]. We will show that in fact every outerplanar map admits a well-orderly tree. It can be computed by the following recursive algorithm *Traversal*.  $H$  is the rooted outerplanar map, and  $r$  is the tail of its root edge.  $\text{Traversal}(H, \emptyset, r)$  returns the well-orderly tree  $T$  of  $H$  rooted at  $r$ , the second parameter is the current set of edges of the tree.

---

**Algorithm 1**  $\text{Traversal}(H, T, u)$ .

---

```

 $C \leftarrow \{(u, v) \in H \mid v \notin T\}$ 
 $T \leftarrow T \cup C$ 
for all edges  $(u, v) \in C$  taken in the clockwise order around  $u$  do
     $T \leftarrow \text{Traversal}(H, T, v)$ 
end for
return  $T$ 

```

---

**Theorem 1** *Every rooted outerplanar map  $H$  has a well-orderly tree  $T$ , computable in linear time, rooted at the tail of the root edge of  $H$ . Moreover, for every node  $u$ , if  $(u, v) \in B_{>}(u)$ , then  $v$  is the next unrelated node with  $u$  in the clockwise preordering of  $T$ . In particular,  $|B_{>}(u)| \leq 1$  for every  $u$ .*

**Proof.** Let  $T$  be the set of edges returned by  $\text{Traversal}(H, \emptyset, r)$  (cf. Algorithm 1), where  $r$  is the tail of the root edge of  $H$ . Let us denote by  $T_i$  and by

$v_i$  respectively the second and the third parameters of the  $i$ th call of *Traversal*. We have  $T_1 = \emptyset$  and  $v_1 = r$ . Note that there are exactly  $n$  calls to *Traversal*, where  $n$  is the number of nodes of  $H$ . By induction,  $T_i$  is a tree with  $i$  nodes, for every  $i \in \{1, \dots, n\}$ . Therefore  $T_n = T$  is a spanning tree of  $H$ . An important observation is that the clockwise preordering of the nodes of  $T$  is precisely  $v_1, \dots, v_n$ , and that  $T_i$  is a subtree of  $T_{i+1}$ , for  $i \in \{1, \dots, n-1\}$ .

This algorithm can be easily implemented to run in  $O(n)$  time. Let us show that the tree  $T$  satisfies the properties of Theorem 1. This is done thanks to the following properties.

**Unrelated edges.** Every edge of  $H$  is either in  $T$ , or an unrelated edge. Indeed, let  $(v_i, v_j)$  be any edge with  $i < j$ . Clearly,  $v_j$  is not an ancestor of  $v_i$ . When  $v_i$  is being treated: 1) if  $v_j \in T_i$ , then  $v_j$  is not a descendant of  $v_i$ , and thus  $(v_i, v_j)$  is an unrelated edge; and 2) if  $v_j \notin T_i$ , then  $v_j$  becomes a child of  $v_i$  in  $T_{i+1}$ , thus an edge of  $T_n$  since  $T_i$  is a subtree of  $T_{i+1}$ .

**Blocks.** The incident edges of  $v_i$  form clockwise around  $v_i$  the four blocks  $B_P(v_i)$ ,  $B_<(v_i)$ ,  $B_C(v_i)$ , and  $B_>(v_i)$ . Since  $T$  is a spanning tree of a planar map, all the edges of  $B_<(v_i)$  are (clockwise) after  $B_P(v_i)$  and before the edges of  $B_>(v_i)$ . Let us show that the edges of  $B_C(v_i)$  are after the edges of  $B_<(v_i)$  and before the edges of  $B_>(v_i)$ .

Let us consider the  $i$ th call of *Traversal*. Let  $(v_i, v_t)$  be the last edge (in clockwise order around  $v_i$ ) toward a node that is before  $v_i$  in  $T_i$ . Let  $v_k$  be the nearest common ancestor of  $v_i$  and  $v_t$ . The path from  $v_k$  to  $v_i$ , the edge  $(v_i, v_t)$  and the path from  $v_t$  to  $v_k$  defines a region of the plane,  $R$ , distinct from the outerface. Since  $H$  is outerplanar there is no node inside the region  $R$ . So all the neighbors of  $v_i$  that are not in  $T_i$  follow the edge  $(v_i, v_t)$  in the clockwise order around  $v_i$ . Hence, the edges of  $B_C(v_i)$  are after the edges of  $B_<(v_i)$ .

The same reasoning can be done to show that the edges of  $B_C(v_i)$  are before the edges of  $B_>(v_i)$ .

**Descendant of the parent.** For every  $(v_i, v_j) \in B_>(v_i)$ ,  $v_i$  is a descendant of the parent of  $v_j$ . Assume  $(v_i, v_j) \in B_>(v_i)$ . In particular  $(v_i, v_j) \notin T$ . When  $v_i$  is being treated, the node  $v_j$  must be in  $T_i$ , otherwise  $v_j$  becomes a child of  $v_i$  in  $T_{i+1}$ , and thus in  $T$ . By construction, the only nodes  $v_j$ 's of  $T_i$  that are after the node  $v_i$  in the prefix clockwise preordering of  $T_i$  are such that their parent is an ancestor of  $v_i$ .

At this point, we have proved that every node in  $T$  is well-orderly, and thus  $T$  is a well-orderly tree of  $H$ .

**Next unrelated node.** Let  $(v_i, v_j) \in B_>(v_i)$ . Let  $v_k$  be the first unrelated node with  $v_i$  clockwise after  $v_i$  in  $T$ . Assume that  $v_j \neq v_k$ , so  $i < k < j$ . Let  $v_t$  be the parent of  $v_j$  in  $T$ . The cycle composed of the path in  $T$  from  $v_t$  to  $v_i$ , and of the edges  $(v_i, v_j)$  and  $(v_j, v_t)$  defines a region of the plane,  $R$ , distinct from the outerface. Because  $i < k < j$ ,  $v_k$  must belong to  $R$  or to its boundary. As  $v_j$  is the only node of the boundary of  $R$  that is unrelated with  $v_i$ ,  $v_k$  must belong to  $R$ : a contradiction with the fact that  $H$  is an outerplanar map. It

follows that  $v_j = v_k$ . Therefore, we have showed that  $(v_i, v_j) \in B_{>}(v_i)$  implies that  $v_j$  is the next unrelated node with  $v_i$  in  $T$ .

This completes the proof of Theorem 1. □

Combining Lemma 1 and Theorem 1, it is clear that an outerplanar map  $H$  has an unique decomposition into a well-orderly tree  $T$ , and a set of edges of the kind  $(u, v) \in B_{>}(u)$ , where  $v$  is the next unrelated node after  $u$  in  $T$ . (Recall that an edge of  $H$  belongs either to  $T$  or to a  $B_{>}$  block). Conversely, given  $T$ , and a piece of information on whether  $B_{>}(u)$  is empty or not for each node  $u$ , one can uniquely determine the corresponding rooted outerplanar map. The coding of the cardinality of each  $B_{>}$  block can be done by coloring the nodes of  $T$  as follows: if  $|B_{>}(u)| = 0$ ,  $u$  is colored white, and if  $|B_{>}(u)| = 1$ ,  $u$  is colored black. Observing that for every node  $u$  of the clockwise last branch of  $T$ ,  $|B_{>}(u)| = 0$ , we obtain:

**Corollary 1** *There is a bijection, computable in linear time, between the  $n$ -node bicolored rooted trees where all the nodes of the last branch (including the root) are colored white, and the  $n$ -node rooted outerplanar maps.*

Recall that a graph (or a map) is  $k$ -connected if  $G$  has more than  $k$  nodes and if, for every subset  $X$  of fewer than  $k$  nodes,  $G \setminus X$  is connected [13]. *Biconnected* is a synonym for 2-connected.

**Theorem 2** *There is a bijection, computable in linear time, between the  $(n-1)$ -node bicolored rooted trees with a white root, all leaves colored in black, and the set of  $n$ -node rooted biconnected outerplanar maps.*

**Proof.** To show the theorem, let show that a outerplanar  $H$  map rooted at  $v$  is biconnected if and only if (1) for each leaf  $u$ , different from the last leaf, of the well-orderly tree of  $H$  rooted at  $v$ ,  $|B_{>}(u)| = 1$  and (2) the depth of the last leaf of  $T$  is 1.

Let  $H$  be an outerplanar map and  $T$  its well-orderly tree. Suppose that  $u$  is a leaf of the tree  $T$  such that  $|B_{>}(u)| = 0$ . All the neighbors of  $u$  belong to the subtree rooted at  $P(u)$  and are before  $u$  in the clockwise prefix order of  $T$ . Let  $u'$  be a node of the subtree rooted at  $P(u)$ . Let  $(u'', u')$  the front edge of  $u''$ . By definition  $u'$  is the first unrelated node of  $u''$  after  $u''$  in the clockwise prefix order of  $T$ . So, either  $u'$  is equal to  $P(u)$  or  $u''$  is in the subtree rooted at  $P(u)$ . Consider the graph  $H'$  obtained from  $H$  when removing  $P(v)$ . If the node  $P(v)$  is not the root of  $T$  then  $H$  is not biconnected. If the node  $P(v)$  is the root of  $T$  and  $u$  is not on the last branch then  $H$  is not biconnected.

Assume now that for all leaves  $u$  (different from the last leaf) of  $T$   $|B_{>}(u)| = 1$  and that the depth of the last leaf is 1. Let  $u_1$  and  $u_2$  be two nodes of  $H$ . W.l.o.g. suppose that  $u_1$  is before  $u_2$  in the prefix clockwise order of  $T$ . To show that  $H$  is biconnected, let us construct a cycle, node disjoint containing  $u_1$  and  $u_2$ . The first part  $C_1$  of this cycle contains the edge  $(l, v)$  where  $l$  is the last leaf of  $T$ , and the path of  $v$  to  $u_1$  in  $T$ . The second part  $C_2$  of the cycle is constructed recursively as follow. Let the variable  $t$  initializes with  $u_1$ . If  $u_2$

does not belong to the subtree of  $T$  rooted at  $t$ , then we add the path from  $t$  to the last leaf  $l$  of the subtree rooted at  $t$  plus the edge  $(l, t') \in B_{>}(l)$  and  $t$  receives now node  $t'$ . If  $u_2$  belongs to the subtree of  $T$  rooted at  $t$ , then we add the path from  $t$  to the last leaf  $l$  of the subtree of  $T$  rooted at  $u_2$  plus the edge  $(l, t') \in B_{>}(l)$  and  $t$  receives now the node  $t'$ .

The construction ends when  $t$  is equal to the last leaf of the tree  $T$ . Clearly the cycle  $C = C_1 \cup C_2$  is node-disjoint and contains the nodes  $u_1$  and  $u_2$ . Hence, there exists two disjoint paths between  $u_1$  and  $u_2$ . This property holds for each pair  $u_1$  and  $u_2$  so the outerplanar map  $H$  is biconnected.  $\square$

Observe that if the well-orderly tree of an outerplanar map  $H$  has its clockwise last leaf of depth  $d$ , then from Corollary 1  $H$  has no more than  $(n - 1) + n - (d + 1) = 2n - 2 - d$  edges. In particular, the depth of the last leaf of the well-orderly tree of any maximal outerplanar map (i.e., having  $2n - 3$  edges) must be 1. As the well-orderly tree is unique, this yields another bijective proof of the following well known result:

**Corollary 2** *There is a bijection, computable in linear time, between maximal  $n$ -node outerplanar maps and planted trees with  $n - 1$  nodes.*

### 3 Enumeration of Outerplanar Maps

Let  $T_{n,d}$  be the number of rooted  $n$ -node plane trees whose clockwise last leaf is of depth  $d$ . These numbers are called the ballot numbers (or Delannoy numbers), and we have [15, 18]:

$$T_{n,d} = \frac{d}{2n - 2 - d} \binom{2n - 2 - d}{n - 1 - d}, \quad \text{for all } n > d > 0.$$

So, there are exactly  $2^{n-1-d}T_{n,d}$  bicolored trees whose all the nodes of the last branch (including the root of the tree) are colored white. Let  $M_n$  be the number of  $n$ -node outerplanar maps. The sequence  $M_n$ , for  $n = 1, 2, \dots$ , is: 1, 1, 3, 13, 67, 381, 2307, 14589, ... (see Figure 3 for all the outerplanar maps of 4 nodes).

**Theorem 3** *For all  $n \geq 2$ ,  $M_n = \sum_{d=1}^{n-1} 2^{n-d-1}T_{n,d}$ , and  $M_n \sim \frac{2^{3n}}{36n\sqrt{\pi n}}$ .*

**Proof.** The enumeration formula for  $M_n$  follows directly from Corollary 1, and of the above formula for  $T_{n,d}$ . Let us compute the asymptotic value of  $M_n$ . Developing the factorials in binomial terms of  $T_{n,d}$ , we have:

$$\frac{T_{n,d}}{T_{n,1}} = d \prod_{j=2}^d \frac{n - j}{2n - 2 - j}.$$

Therefore,

$$\frac{M_n}{2^{n-2}T_{n,1}} = \sum_{d=1}^{n-1} \frac{d}{2^{d-1}} \prod_{j=2}^d \frac{n - j}{2n - 2 - j} = \sum_{d=1}^{n-1} u_{n,d},$$

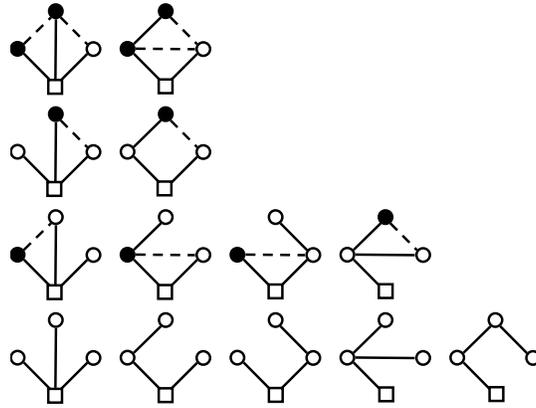


Figure 3: The 13 rooted outerplanar maps with 4 nodes.

with

$$u_{n,d} = \frac{d}{2^{d-1}} \prod_{j=2}^d \frac{n-j}{2n-2-j}.$$

For every  $n$ , let us set  $v_d(n) = u_{n,d}$  if  $d < n$ , and  $v_d(n) = 0$  if  $d \geq n$ . As  $(n-j)/(2n-2-j) \leq 1/2$ ,  $|v_d(n)| \leq d/2^{d-1}(1/2)^{d-1} = d/2^{2d-2}$ , for all  $d$  and  $n$ . Let  $w_d = d/2^{2d-2}$ . We have:

$$\sum_{d=1}^{\infty} w_d = \lim_{n \rightarrow \infty} \sum_{d=1}^n w_d = \lim_{n \rightarrow \infty} \left( \frac{16}{9} - \frac{4}{3} \left( \frac{1}{4} \right)^n \left( n + \frac{4}{3} \right) \right) = \frac{16}{9}.$$

Because the series  $w_d$  is convergent, the series  $v_d(n)$  is normally convergent. As  $\lim_{n \rightarrow \infty} v_d(n) = w_d$ , we have:

$$\lim_{n \rightarrow \infty} \sum_{d=1}^{n-1} u_{n,d} = \lim_{n \rightarrow \infty} \sum_{d=1}^{\infty} v_d(n) = \sum_{d=1}^{\infty} \lim_{n \rightarrow \infty} v_d(n) = \sum_{d=1}^{\infty} w_d = \frac{16}{9}.$$

We have:

$$T_{n,1} = \frac{1}{2n-3} \binom{2n-3}{n-2} = \frac{1}{n} \binom{2(n-2)}{n-2} \sim \frac{2^{2n}}{16n\sqrt{\pi n}}.$$

The result follows. □

Narayana numbers are related with biconnected outerplanar maps. The Narayana numbers [21] count rooted plane trees with  $n$  nodes and  $\ell$  leaves:

$$Y_{n,\ell} = \frac{1}{n} \binom{n}{\ell} \binom{n-2}{n-\ell-2}, \quad \text{for all } n > \ell > 0.$$

Biconnected outerplanar graphs can be seen as dissections of a convex polygon, and their number can be counted by Schröder numbers. From Theorem 2, we have:

**Corollary 3** *The number of rooted biconnected outerplanar maps with  $n \geq 3$  nodes is  $\sum_{\ell=2}^{n-2} 2^{n-1-\ell} Y_{n-1,\ell}$ .*

**Theorem 4** *The number  $M_{n,m}$  of rooted outerplanar maps with  $n \geq 3$  nodes and  $m \geq n - 1$  edges is:*

$$M_{n,m} = \sum_{d=1}^{2n-2-m} \binom{n-d-1}{m-n+1} T_{n,d}.$$

**Proof.** Outerplanar maps with  $n$  nodes and  $m$  edges are in bijection with bicolored trees with white nodes on the last branch and  $m - n + 1$  black nodes (see Corollary 1). On each tree of  $T_{n,d}$ , there are  $\binom{n-d-1}{m-n+1}$  possible ways to color  $m - n + 1$  nodes that are not on the last branch in black.  $\square$

Let  $\mathcal{M}_n$  denote the set of  $n$ -node outerplanar maps, counted by  $M_n$ , and let  $\mathcal{M}_{n,m}$  denote the  $n$ -node  $m$ -edge outerplanar maps, counted by  $M_{n,m}$ . Thanks to Corollary 1, for each  $H \in \mathcal{M}_n$ , we can uniquely associate a number  $d$  that is the depth of the last leaf of the bicolored tree  $T$  in bijection with  $H$ . Let  $\bar{d}$  be the average value of  $d$  in  $\mathcal{M}_n$ .

**Lemma 2** *For every  $n$ ,  $\bar{d} < 3$ .*

**Proof.** Let us prove that

$$\bar{d} = \frac{\sum_{d=1}^{n-1} d 2^{n-1-d} T_{n,d}}{\sum_{d=1}^{n-1} 2^{n-1-d} T_{n,d}} \leq \frac{\sum_{d=1}^{n-1} d T_{n,d}}{\sum_{d=1}^{n-1} T_{n,d}}. \quad (1)$$

Take the positive functions  $f$  and  $g$ . For  $i, j \in \mathbb{N}$  and  $f$  monotonic decreasing, we show that the following expression is negative:

$$\begin{aligned} & \frac{\sum_i i \cdot f(i)g(i)}{\sum_i f(i)g(i)} - \frac{\sum_i i \cdot g(i)}{\sum_i g(i)} \\ = & \frac{(\sum_i i \cdot f(i)g(i))(\sum_j g(j)) - (\sum_i i \cdot g(i))(\sum_j f(j)g(j))}{(\sum_i f(i)g(i))(\sum_j g(j))}. \end{aligned}$$

Consider only the numerator  $N$  of the previous function.  $N$  can be rewritten as:

$$\begin{aligned} & (\sum_i i \cdot f(i)g(i)) (\sum_j g(j)) - (\sum_i i \cdot g(i)) (\sum_j f(j)g(j)) \\ = & \sum_i \sum_j i \cdot f(i)g(i)g(j) - i \cdot g(i)f(j)g(j) \\ = & \sum_i \left( \sum_{j < i} i \cdot g(i)g(j)(f(i) - f(j)) + \sum_{j > i} i \cdot g(i)g(j)(f(i) - f(j)) \right) \\ = & \sum_i \left( \sum_{j < i} i \cdot g(i)g(j)(f(i) - f(j)) + \sum_{j < i} j \cdot g(j)g(i)(f(j) - f(i)) \right) \\ = & \sum_i \sum_{j < i} (i - j) \cdot g(i)g(j)(f(i) - f(j)). \end{aligned}$$

If  $f$  is a monotonically decreasing function, the previous expression is negative. Since  $f(d) = 2^{n-1-d}$  is monotonic decreasing, Inequality (1) is proved.

Observe that the right term of Inequality (1) is the average depth  $\bar{d}_t$  of the last leaf of an  $n$ -node planted tree. Let  $\mathcal{T}_n$  denote the set of  $n$ -node planted trees. Let us show that  $\bar{d}_t \leq 3\frac{n-1}{n+1}$ .

Let us consider the sequence of bijections of the following diagram:

$$\mathcal{T}_n \leftrightarrow \mathcal{D}_{2n-2} \xrightarrow{\sim} \mathcal{D}_{2n-2} \leftrightarrow \mathcal{T}_n \leftrightarrow \mathcal{G}_{n+1}.$$

where 1)  $\mathcal{D}_{2n}$  is the set of Dyck words<sup>2</sup> of length  $2n$ , 2)  $\mathcal{G}_{n+1}$  is the set of rooted triangulations of an  $(n + 1)$ -gon.

Almost all these bijections are well-known. Let us remind some facts and a new bijection between  $\mathcal{T}_n$  and  $\mathcal{G}_{n+1}$ . Consider the bijections from the left to the right. In the usual edge encoding  $w \in \mathcal{D}_{2n-2}$  of a ordered rooted tree  $t \in \mathcal{T}_n$ , the *clockwise prefix coding*<sup>3</sup>, the length of the last branch of  $t$  corresponds to the number of consecutive closed parentheses at the end of  $w$ . The operation  $\sim$  consists in reading  $w$  from the right to the left and substituting each open (respectively closed) parenthesis by a closed (respectively open) one. One can easily check that the result  $\tilde{w}$  is also a Dyck word and the distribution of the number of consecutive open parentheses in the beginning of Dyck words is equal to the distribution of closed parentheses at the end of Dyck words.

Take another encoding of  $t \in \mathcal{T}_n$ , the *degree encoding*: start with the empty Dyck word, perform a prefix traversal of the first  $n - 1$  nodes of  $t$  appending for each node having  $i$  children the pattern  $\underbrace{((\dots (}_i)$ . (The last node of  $t$  being a leaf, we do not need to code  $)$  for it.) The final word is of length  $2n - 2$ . The number of consecutive open parentheses in the beginning of  $\tilde{w}$  represents now the degree of the root of  $t$ .

By Corollary 2, there is a bijection between  $\mathcal{G}_{n+1}$  and  $\mathcal{T}_n$ . Let us remind some details of the bijection. If we apply Algorithm *Traversal* to a rooted triangulation of an  $(n + 1)$ -gon  $g$ , the result of the algorithm is of course a bicolored tree  $t'$  of  $n + 1$  nodes. Through this new bijection, the degree of the root of  $g$  and of  $t'$  are the same. Removing the last branch of length 1 and ignoring the color of the nodes of  $t'$ , we obtain a tree  $t \in \mathcal{T}_n$ . The degree of the root of  $t$  is equal to the degree of the root of  $t'$  minus one.

Through these bijections, the distribution of the degree of the root in  $\mathcal{G}_{n+1}$  is in bijection with the distribution of the length of the last branch in  $\mathcal{T}_n$  minus one.

Let  $\bar{d}_g$  be the average degree of the root in  $\mathcal{G}_{n+1}$ .  $\bar{d}_t = \bar{d}_g - 1 = 2\frac{2n-1}{n+1} - 1 = 3\frac{n-1}{n+1}$ . □

Let  $\bar{m}_n$  be the average number of edges of the maps belonging to  $\mathcal{M}_n$ . From Theorems 3, 4, and Lemma 2, we can deduce the following result:

<sup>2</sup>Dyck words or words of balanced parentheses are the set of words belonging to  $\{(,)\}^*$  with the same amount of open and closed parentheses and so that at least half of the letters of any prefix are a occurrence of open parentheses.

<sup>3</sup>Each edge encountered for the first time is coded by an open parenthesis and otherwise by a closed one.

**Corollary 4** For every  $n$ ,  $\frac{3}{2}n - 3 < \overline{m}_n \leq \frac{3}{2}n - 2$ .

**Proof.** Let  $\overline{m}_{n,d}$  be the average number of edges for given  $d$ .

$$\overline{m}_{n,d} = \frac{\sum_{m=n-1}^{2n-d-2} m \binom{n-d-1}{m-n+1} T_{n,d}}{2^{n-d-1} T_{n,d}}.$$

Taking  $k = m - n + 1$  and using the well-known binomial identities

$$\sum_{d=0}^n \binom{n}{d} = 2^n \quad \text{and} \quad \sum_{d=0}^n d \binom{n}{d} = n2^{n-1},$$

we have

$$\begin{aligned} \overline{m}_{n,d} &= \frac{\sum_{k=0}^{n-d-1} (n+k-1) \binom{n-d-1}{k}}{2^{n-d-1}} \\ &= \frac{(n-1) \sum_{k=0}^{n-d-1} \binom{n-d-1}{k} + \sum_{k=0}^{n-d-1} k \binom{n-d-1}{k}}{2^{n-d-1}} \\ &= \frac{(n-1)2^{n-d-1} + (n-d-1)2^{n-d-2}}{2^{n-d-1}} \\ &= \frac{3n-d-3}{2}. \end{aligned}$$

By definition,

$$\begin{aligned} \overline{m}_n &= \frac{\sum_{d=1}^{n-1} \overline{m}_{n,d} M_{n,d}}{M_n} \\ &= \frac{\sum_{d=1}^{n-1} (3n-d-3) M_{n,d}}{2M_n} \\ &= \frac{(3n-3) \sum_{d=1}^{n-1} M_{n,d} - \sum_{d=1}^{n-1} d M_{n,d}}{2M_n} \\ &= \frac{3n-3}{2} - \frac{\bar{d}}{2}. \end{aligned}$$

From Lemma 2,  $1 \leq \bar{d} < 3$  and  $\frac{3n}{2} - 2 \geq \overline{m}_n > \frac{3n}{2} - 3$ . □

## 4 Coding Supporting Adjacency, Degree and Neighbors Queries

From Corollary 1, every outerplanar map with  $n$  nodes can be represented by a bicolored tree with  $n$  nodes. A standard coding for  $n$ -node planted trees uses  $2n - 4$  bits if  $n \geq 2$ , and the colors can be stored using  $n - 2$  bits at most, observing that the color associated to the last branch (containing the last leaf and the root) is known (white). For the tree encoding, we use a clockwise depth-first traversal of the tree, each edge being traversed twice starting from the root.

During the traversal, we output “1” if the edge is traversed for the first time, and “0” otherwise. This leads to a  $2(n - 1)$ -bit encoding. Actually, if the tree has at least one edge ( $n \geq 2$ ) two bits can be saved observing that the previous coding always starts with a “1” and ends with a “0”.

This leads to a  $(3n - 6)$ -bit encoding. By Lemma 3, the length of this coding is asymptotically optimal, up to an additive factor of  $O(\log n)$  bits.

**Theorem 5** *Every  $n$ -node rooted outerplanar map or every outerplanar graph,  $n \geq 2$ , can be coded (and decoded) in  $O(n)$  time and using a representation of at most  $3n - 6$  bits.*

In the following, we show how to extend this coding with an additional  $o(n)$  bits (still constructible in linear time) so that the data structure supports adjacency and degree queries in worst-case constant time, and neighbors queries in  $O(d)$  time. For that we present below efficient well-known data structures for binary strings and balanced string of parentheses.

#### 4.1 Constant time queries in strings of parentheses

Let  $S$  be a string of symbols defined over a given alphabet of constant size. We denote by  $S[i]$  the  $i$ th symbol of  $S$ ,  $i \geq 1$ . Let  $\text{select}(S, i, \star)$  be the position of the  $i$ th  $\star$  in  $S$ . Let  $\text{rank}(S, j, \star)$  be the number of symbols  $\star$  before or at the  $j$ th position of  $S$ . That is if  $\text{select}(S, i, \star) = j$  then  $\text{rank}(S, j, \star) = i$ .

Now, let  $S$  be a string of open and closed parentheses, i.e.,  $\star \in \{ (, ) \}$ . Two parentheses  $S[i] = ($  and  $S[j] = )$  match if  $i < j$  and the difference of the number of open and closed parentheses between them is null. The string  $S$  is *balanced* if for each parenthesis, there is a matching parenthesis. Let  $\text{match}(S, i)$  be the position in  $S$  of the matching parenthesis of  $S[i]$ .  $S[k]$  is *enclosed* by  $S[i]$  and  $S[j]$  if  $i < k < j$ . Let  $\text{enclose}(S, i_1, i_2)$  be the closest matching parenthesis pair  $(j_1, j_2)$  that encloses  $S[i_1]$  and  $S[i_2]$ . Finally, let  $\text{wrapped}(S, j)$  denote twice the number of pairs of matching parentheses  $(i_1, i_2)$  such that  $\text{enclose}(S, i_1, i_2) = (j, \text{match}(S, j))$ .

The following results are valid in the word-RAM model, in which standard arithmetic operations on binary words of length  $O(\log n)$  can be done in constant time.

**Lemma 3 ([20, 9])** *For every balanced string  $S$  of parentheses (or a binary string) of length  $O(n)$ , the operations  $\text{select}$ ,  $\text{rank}$ ,  $\text{match}$ ,  $\text{enclose}$ , and  $\text{wrapped}$  can be done in worst-case constant time using an auxiliary table of  $o(|S|)$  bits and  $O(|S|)$  preprocessing time.*

Actually, Lemma 3 holds for the operations  $\text{select}$  and  $\text{rank}$ , even if  $S$  is not balanced.

#### 4.2 Coding of outerplanar maps

We can associate to any planted tree  $T$  a balanced string of parentheses (or Dyck word) as follows: during a clockwise depth-first traversal of  $T$  starting at the

root, if an edge is traversed for the first time, output an open parenthesis and otherwise output a closed parenthesis. For convenience, an open parenthesis (respectively a closed parenthesis) is added at the beginning (respectively at the end) of the output string. One can regard this latter transformation as an extra edge entering in the root of  $T$ . If  $T$  has  $n$  nodes, the final string of parentheses contains  $2n$  symbols as each of  $T$  (plus the extra edge) is traversed twice. The final string is called the *clockwise prefix coding* of  $T$ .

Consider  $T$  an  $n$ -node planted tree, and its clockwise prefix coding  $S_T$ . Let  $v_1, \dots, v_n$  be the clockwise preordering of the nodes of  $T$ . To support efficiently queries on  $T$ , we label the node  $v_i$  by its index  $i$ , so that an adjacency query between the nodes  $v_i$  and  $v_j$  is simply the pair  $(i, j)$ . By construction of the clockwise prefix coding, the  $i$ th open parenthesis corresponds to the edge of  $T$  entering in  $v_i$ . And, the matching parenthesis of the  $i$ th open parenthesis corresponds to the edge of  $T$  leaving  $v_i$ . In other words, each node  $v_i$  can be seen as a pair  $(i, j)$  of matching parentheses, i.e., such that  $S_T[i]$  matches with  $S_T[j]$  in  $S_T$ . See Example 1.

Lemma 3 and the clockwise prefix coding lead to two useful operations for the tree that can be supported in constant time: adjacency and degree.

Two nodes are adjacent in  $T$  if one of them is the parent of the other one. One can determine the parent of a node  $v_i$  finding the position  $p$  of the  $i$ th open parenthesis using  $p = \text{select}(S_T, i, \text{()})$ , and the position of the closest enclosing pair  $(j_1, j_2)$  of  $(p, \text{match}(S_T, p))$  with  $(j_1, j_2) = \text{enclose}(S_T, p, \text{match}(S_T, p))$ . Then, the parent of  $v_i$  is  $v_j$  where  $j = \text{rank}(S_T, j_1, \text{()})$ .

As for the degree, one can check that the number of children of  $v_i$  is  $\frac{1}{2}\text{wrapped}(S_T, \text{select}(S_T, i, \text{()}))$ . We just have to add 1 if  $v_i$  is not the root, i.e., if  $i \neq 1$ .

From the previous explanations, one can claim the following lemma:

**Lemma 4 ([9])** *Let  $T$  be an  $n$ -node planted tree and its clockwise prefix coding  $S_T$ . The parent, degree and adjacency of a node can be answered in constant time using an auxiliary table of  $o(|S_T|)$  bits and an  $O(n)$  preprocessing time.*

We encode an outerplanar map  $H$  with two binary strings:

1.  $S_T$  of length  $2n$ : the clockwise prefix coding of the well-orderly tree  $T$  of  $H$ ;
2.  $S_B$  of length  $n$ : a binary string such that  $S_B[j] = |B_{>}(v_i)|$ , where  $v_i$  is the node corresponding to the  $j$ th closed parenthesis of  $S_T$ .

**Remark 1** *Given  $j$ ,  $i$  can be obtained by:*

$$i = \text{rank}(S_T, \text{match}(S_T, \text{select}(S_T, j, \text{()})), \text{()}).$$

*Observe that if  $j > n - 2$ , then  $v_i$  belongs to the last branch of  $T$ , and thus  $|B_{>}(v_i)| = 0$ . Thus,  $S_B$  can reduce to the first  $n - 2$  entries.*

*To know if node  $v_i$  has a front edge, that is to perform  $|B_{>}(v_i)|$  given  $i$ , one can compute  $j = \text{rank}(S_T, \text{match}(S_T, \text{select}(S_T, i, \text{()})), \text{()})$ , and return  $S_B[j]$ .*

The sequence of nodes  $v_{i_1}, v_{i_2}, \dots, v_{i_j}$  is a *branch* of  $T$  if  $v_{i_j}$  is a leaf, and if for every  $t \in \{1, \dots, j-1\}$ ,  $v_{i_{t+1}}$  is the clockwise last child of  $v_{i_t}$ . The last branch consists therefore of all the nodes of the path between the root and the clockwise last leaf of  $T$ . The branches partition the nodes of  $T$  and there is exactly one branch per leaf. One can check that a branch of  $T$  corresponds to a maximal block of closed parentheses in  $S_T$ .

**Example 1** *The outerplanar map of Figure 2 can be encoded by the following two strings:*

$S_T = ((((((((())((())())((())())))))))$  and  $S_B = 101110111010000$ .

*The sequence of nodes  $v_3, v_8, v_{10}$  is a branch. To know if node  $v_3$  has a front edge, do the following sequence of operations:  $p = \text{select}(S_T, 3, ()) = 3$ ,  $\text{match}(S_T, p) = 18$ ,  $\text{rank}(S_T, 18, ) = 8$ . Since  $S_B[8] = 1$ , node  $v_3$  has a front edge.*

**Lemma 5** *Let  $v_{i_1}, \dots, v_{i_j}$  be a branch of  $T$ . We have*

$$|B_{<}(v_{i_{j+1}})| = \sum_{t=1}^j |B_{>}(v_{i_t})|.$$

*Moreover,  $|B_{<}(v_{i_{j+1}})|$  can be computed in constant time.*

**Proof.** Clearly, the next unrelated node with any node  $v_i$  is the node  $v_{r+1}$  such that  $v_r$  is the leaf of the branch of  $v_i$ . In particular, the next unrelated node of every node  $v_{i_t}$ ,  $t \in \{1, \dots, j\}$ , is  $v_{i_j+1}$ . Let  $k = i_j + 1$ . Let  $v_i$  be any unrelated node with  $v_k$  and with  $i < i_1$ . The next unrelated node of  $v_i$  is a node  $v_{r+1}$  where  $v_r$  is the leaf of the branch of  $v_i$ . So we have  $r < i_1$ , that implies  $v_k \neq v_{r+1}$ . It follows that the only nodes whose the next unrelated node is  $v_k$  are the nodes  $v_{i_1}, \dots, v_{i_j}$ . It follows that  $|B_{<}(v_{i_j+1})| = \sum_{t=1}^j |B_{>}(v_{i_t})|$ .

To compute  $|B_{<}(v_{i_j+1})|$ , we need to find in  $S_T$  the positions corresponding to the beginning and the end of the branch. A branch corresponds to a maximal sequence of closed parentheses in  $S_T$ . The position  $p$  of  $v_k$  in  $S_T$  is  $p = \text{select}(S_T, k, ())$ . The last closed parenthesis of branch  $b$  corresponding to  $v_{i_1}$  is at position  $p-1$  in  $S_T$ . To find the position  $p'$  of the closed parenthesis of  $v_{i_j}$ , we compute  $p' = \text{select}(S_T, k-1, ())$ . The cardinality of  $B_{<}(v_k)$  is the number of ones in  $S_B$  from position  $p_1 = \text{rank}(S_T, p', )$  to position  $p_2 = \text{rank}(S_T, p-1, )$ . Let  $r_1$  (respectively  $r_2$ ) be the number of black nodes before the  $p_1$ -th (respectively  $p_2$ -th) node in the postfix order of  $T$ :  $r_1 = \text{rank}(S_B, p_1, 1)$  and  $r_2 = \text{rank}(S_B, p_2, 1)$ . Thus,  $|B_{<}(v_k)| = r_2 - r_1$ .  $\square$

**Theorem 6** *Every rooted outerplanar map with  $n$  nodes admits a  $3n + o(n)$ -bit encoding, constructible in linear time, such that adjacency and degree queries can be computed in worst-case constant time and the neighbors of a  $d$ -degree node can be determined in worst-case  $O(d)$  time.*

**Proof.** We proposed a  $3n$ -bit encoding for a rooted outerplanar map. Adding auxiliary tables of size  $o(n)$ , by Lemmas 3 and 4, the parent and the degree of  $v_i$  in  $T$  can be obtained in  $O(1)$  time.

---

**Algorithm 2** *Adjacency*( $S_T, S_B, i, j$ ), assuming  $i < j$ 


---

```

if  $i$  is the parent of  $j$  in  $T$  then
  return true
else
   $i' \leftarrow \text{match}(S_T, \text{select}(S_T, i, ()))$ 
   $r \leftarrow \text{rank}(S_T, i', ())$ 
  if  $r \neq j - 1$  then
    return false
  else
    return  $S_B[r]$ 
  end if
end if

```

---

**Adjacency:** it remains to check the adjacency for the edges of  $H$  that does not belong to  $T$ . Nodes  $v_i$  and  $v_j$ , with  $i < j$ , are adjacent in  $H$  if  $v_j$  is the next unrelated node after  $v_i$  (that is  $j = r - 1$  where  $v_r$  is the leaf of the branch of  $v_i$ ) and if  $|B_{>}(v_i)| = 1$ . See Algorithm 2 for details.

**Degree:** the degree of node  $v_i$  is the sum of the degree within tree  $T$  (see Lemma 4), the cardinality of  $B_{<}(v_i)$  (see Lemma 5) and of  $B_{>}(v_i)$ . Recall that if  $v_{i-1}$  is not a leaf,  $B_{<}(v_i) = \emptyset$ .

**Neighbors:** The list  $L$  of neighbors  $v_j$  of a node  $v_k$  contains the neighbors  $L_T$  within the tree  $T$  and the unrelated neighbors  $L_R$ .

The list  $L_T$  can be found in  $O(L_T)$  time as in [20] in which the  $d$  neighbors of a node within a tree is computed in  $O(d)$  with a similar encoding.

The list  $L_R$  contains nodes reached either by the front edge or by the edges of  $B_{>}(v_k)$ .

If  $v_k$  has a front edge leading to  $v_j$ ,  $v_j$  is the first unrelated node after  $v_k$  can be computed in constant time as follows:

$$j = \text{rank}(S_T, \text{match}(S_T, \text{select}(S_T, k, ()), ()), ()) + 1.$$

If  $B_{<}(v_k) \neq \emptyset$  (see Lemma 5 and above), then the edge  $(v_k, v_j) \in B_{<}(v_k)$  if and only if  $v_j$  has a front edge and belongs to the branch of the leaf  $v_{k-1}$ . Using the same notation as in the proof of Lemma 5, the labels of these nodes can be computed as follows: for each  $r$ ,  $r_1 + 1 \leq r \leq r_2$  we get the label  $j$  of the  $r$ -th node having a front edge in the postfix order in constant time:  $j = \text{rank}(S_T, \text{match}(S_T, \text{select}(S_T, \text{select}(S_B, r, 1), ())), ())$ .  $\square$

Starting from a connected outerplanar graph, we can compute a rooted outerplanar map using the algorithm presented in [10]. So the previous results on outerplanar maps can also be applied to outerplanar graphs.

## 5 Uniform Random Generation

To randomly generate an outerplanar map, one can randomly generate a bicolored tree where the nodes of the last branch are colored white. Thanks to Corollary 1, one can then construct in linear time the corresponding random outerplanar map.

**Theorem 7** *A rooted outerplanar map with  $n$  nodes or with  $n$  nodes and  $m$  edges can be generated uniformly at random in  $O(n)$  expected time.*

**Proof.** Let  $\mathcal{B}_{n,b}$  be the set of all bicolored rooted trees with  $n$  nodes such that:

1. the root is colored white;
2. the clockwise last leaf is colored white; and
3. there are exactly  $b$  nodes colored black.

Let  $\mathcal{B}_n = \bigcup_{b=0}^{n-2} \mathcal{B}_{n,b}$ . From the bijection between outerplanar maps and some bicolored trees (cf. Corollary 1), the outerplanar maps with  $n$  nodes are in bijection with a subset, say  $\tilde{\mathcal{M}}_n$ , of bicolored trees. Clearly,  $\tilde{\mathcal{M}}_n \subset \mathcal{B}_n$ . Similarly, the outerplanar maps with  $n$  nodes and  $m$  edges are in bijection with a subset, say  $\tilde{\mathcal{M}}_{n,m}$ , of bicolored trees. The trees of  $\tilde{\mathcal{M}}_{n,m}$  have exactly  $m - (n - 1)$  black nodes, so  $\tilde{\mathcal{M}}_{n,m} \subset \mathcal{B}_{n,m-n+1}$ .

Our algorithm is an accept-reject algorithm: it consists in repeating a uniformly generation of an element of  $\mathcal{B}_n$  (or of  $\mathcal{B}_{n,m-n+1}$ ) until we get an element of  $\tilde{\mathcal{M}}_n$  (or of  $\tilde{\mathcal{M}}_{n,m}$ ). This clearly provides a uniform random generation on the set  $\tilde{\mathcal{M}}_n$  (or on  $\tilde{\mathcal{M}}_{n,m}$ ), and thus on the corresponding outerplanar maps.

A rooted tree can be generated in linear time using for example the Arnold and Sleep's algorithm [2]. Each of the  $n - 2$  nodes is colored black with probability  $1/2$  (recall that the root and the last leaf are forced to be colored white by definition of  $\mathcal{B}_n$ ). This provides an  $O(n)$  time algorithm to generate an element of  $\mathcal{B}_n$ .

To generate an element of  $\mathcal{B}_{n,m-n+1}$ , we have to select exactly  $b = m - n + 1$  black nodes among  $n - 2$  (the root and the last leaf are forced to be colored white). Selecting  $b$  elements among  $n - 2$  can be done in  $O(n)$  time [22] (the procedure uses  $O(n)$  arithmetic operations on  $O(\log n)$  bit integers). This provides an  $O(n)$  time algorithm to generate an element of  $\mathcal{B}_{n,m-n+1}$ .

Testing whether  $T \in \mathcal{B}_n$  belongs to  $\tilde{\mathcal{M}}_n$  (or whether  $T \in \mathcal{B}_{n,m-n+1}$  belongs to  $\tilde{\mathcal{M}}_{n,m}$ ) clearly takes  $O(n)$  time: it suffices to test whether all the inner nodes of the last branch are white or not.

As the bijection between bicolored trees and maps can be obtained in linear time, it remains to show that the average number of rejects in the above procedure is bounded by a constant.

Observe that every tree  $T \in \mathcal{B}_n$  whose last leaf is of depth 1 belongs to  $\tilde{\mathcal{M}}_n$  and  $\tilde{\mathcal{M}}_{n,m-n+1}$ . Thus the probability to accept a tree  $T \in \mathcal{B}_n$  in  $\tilde{\mathcal{M}}_n$  (or in

$\tilde{\mathcal{M}}_{n,m-n+1}$ ) is at least :

$$\frac{T_{n,1}2^{n-2}}{|\mathcal{B}_n|} = \frac{T_{n,1}2^{n-2}}{\sum_{d=1}^{n-1} T_{n,d}2^{n-2}} = \frac{c_{n-2}}{c_{n-1}} = \frac{n}{4n-6} > \frac{1}{4}.$$

To see the above inequality, we use the fact that  $\sum_{d=1}^{n-1} T_{n,d} = c_{n-1}$  and  $T_{n,1} = c_{n-2}$ , where  $c_n = \frac{1}{n+1} \binom{2n}{n}$  is the  $n$ th Catalan number counting the number of  $n$ -edge rooted trees. It follows that the average number of rejects is at most 4, thereby completes the proof.  $\square$

The algorithm presented in the proof of Theorem 7 can be enhanced into an anticipated-reject version. In this version, the bicolored tree is constructed from the end (so from the last branch). As soon as an inner black node appears in the last branch, we reject. The advantage of this version is that fewer random bits are needed to generate an outerplanar map since the expected length of the last branch is  $O(1)$ , so only  $O(1)$  bits are consumed before acceptance of the whole bicolored tree. An implementation of the enhanced algorithm is available in the PIGALE Library (<http://pigale.sourceforge.net/>), and an experiments of this algorithm is given in Figure 4.

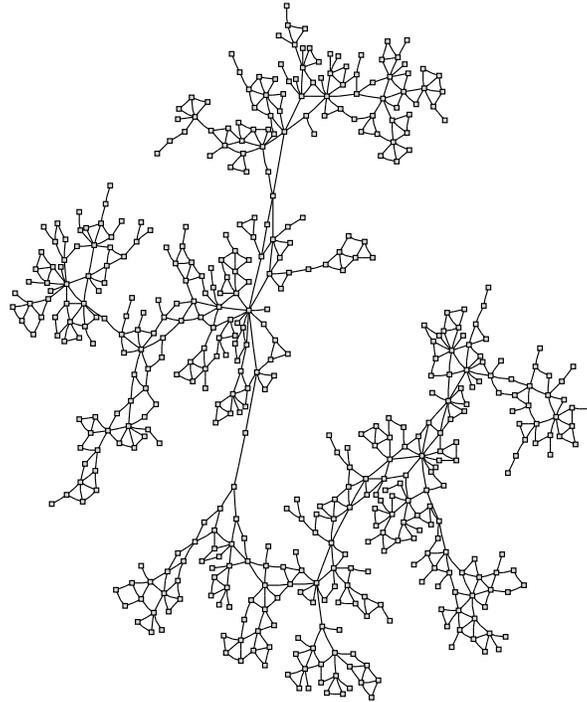


Figure 4: A random outerplanar map with 500 nodes, drawn using a spring embedding algorithm preserving embedding.

## References

- [1] L. Alonso, J. L. Rémy, and R. Schott. A linear-time algorithm for the generation of trees. *Algorithmica*, 17(2):162–182, 1997.
- [2] D. B. Arnold and M. R. Sleep. Uniform random generation of balanced parenthesis strings. *ACM Trans. Programming Languages and Systems*, 2(1):122–128, 1980.
- [3] E. Barucci, A. del Lungo, and E. Pergola. Random generation of trees and other combinatorial objects. *Theoretical Computer Science*, 218(2):219–232, 1999.
- [4] T. Bilski. Embedding graphs in books: A survey. *IEE Proceedings-E*, 139(2):134–138, Mar. 1992.
- [5] M. Bodirsky and M. Kang. Generating random outerplanar graphs. In *1<sup>st</sup> Workshop on Algorithms for Listing, Counting, and Enumeration (AL-ICE)*, Jan. 2003. To appear in *Journal of Combinatorics, Probability and Computing*.
- [6] N. Bonichon, C. Gavaille, and N. Hanusse. An information-theoretic upper bound of planar graphs using triangulation. Technical report, LaBRI Research Report RR-1279-02, 2002.
- [7] N. Bonichon, C. Gavaille, and N. Hanusse. An information-theoretic upper bound of planar graphs using triangulation. In *20<sup>th</sup> Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2607 of *Lecture Notes in Computer Science*, pages 499–510. Springer, Feb. 2003.
- [8] G. Chartrand and F. Harary. Planar permutation graphs. *Ann. Inst. Henry Poincaré, Sec. B3*, pages 433–438, 1967.
- [9] Y.-T. Chiang, C.-C. Lin, and H.-I. Lu. Orderly spanning trees with applications to graph encoding and graph drawing. In *12<sup>th</sup> Symposium on Discrete Algorithms (SODA)*, pages 506–515. ACM-SIAM, Jan. 2001. To appear in *SIAM Journal on Computing*.
- [10] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa. A linear algorithm for embedding planar graphs using pq-trees. *Journal of Computer and System Sciences*, 30(1):54–76, 1985.
- [11] R. Cori and A. Machi. Maps, hypermaps and their automorphisms: a survey i, ii, iii. *Expo. Math.*, 10:403–467, 1992.
- [12] A. Denise and P. Zimmermann. Uniform random generation of decomposable structures using floating-point arithmetic. *Theoretical Computer Science*, 218:233–248, 1999.
- [13] R. Diestel. *Graph Theory (second edition)*, volume 173 of *Graduate Texts in Mathematics*. Springer, Feb. 2000.

- [14] P. Epstein and J.-R. Sack. Generating triangulations at random. *ACM Trans. Model. and Comput. Simul.*, 4:267–278, 1994.
- [15] W. Feller. *An Introduction to Probability Theory and its Applications*, volume 1. John Wiley & Sons, 1968.
- [16] P. Flajolet and M. Noy. Analytic combinatorics of non-crossing configurations. *Discrete Mathematics*, 204:203–229, 1999.
- [17] M. Goldwurm. Random generation of words in an algebraic language in linear binary space. *Information Processing Letters*, 54:229–233, 1995.
- [18] G. Kreweras. Sur les éventails de segments. *Cahiers du Bureau Universitaire de Recherche Opérationnelle*, 15:1–41, 1970.
- [19] S. L. Mitchell. Linear algorithms to recognize outerplanar and maximal outerplanar graphs. *Information Processing Letters*, pages 229–232, 1979.
- [20] J. I. Munro and V. Raman. Succinct representation of balanced parentheses, static trees and planar graphs. *SIAM Journal on Computing*, 31(3):762–776, 2001.
- [21] T. V. Narayana. A partial order and its application to probability. *Sankhya*, 21:91–98, 1959.
- [22] H. Wilf. A unified setting for sequencing, ranking, and selection algorithms for combinatorial objects. *Advances in Mathematics*, 24:281–291, 1977.