Growing random uniform d-ary trees

Jean-François Marckert

CNRS, LaBRI, Université de Bordeaux

Abstract

Let $\mathcal{T}_d(n)$ be the set of d-ary rooted trees with n internal nodes. We give a method to construct a sequence $(\mathbf{t}_n, n \geq 0)$ where, for any $n \geq 1$, $\mathbf{t}_n$ has the uniform distribution in $\mathcal{T}_d(n)$, and $\mathbf{t}_n$ is constructed from $\mathbf{t}_{n-1}$ by the addition of a new node, and a rearrangement of the structure of $\mathbf{t}_{n-1}$. This method is inspired by Rémy's algorithm which does this job in the binary case, but it is different from it. This provides a method for the random generation of a uniform d-ary tree in $\mathcal{T}_d(n)$ with a cost linear in n.

1 Introduction

Notation: in the paper, we denote by $[\![a, b]\!]$ the ordered list of integers belonging to $[a, b] \cap \mathbb{Z}$.

Rooted planar trees are often represented as on Figure 1: there is a root, and the children of the nodes are distinguishable.

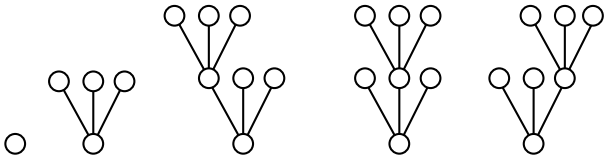

Figure 1: The first tree is the single element of $\mathcal{T}_3(0)$, the next one, the single element of $\mathcal{T}_3(1)$ and the next tree, the three elements of $\mathcal{T}_3(2)$.

A d-ary tree is a rooted planar tree in which all nodes have either d children or none. Nodes with degree d are called *internal nodes*, the other ones are called *leaves*. Let $\mathcal{T}_d(n)$ be the set of d-ary trees with n internal nodes: $\mathcal{T}_2(n)$ is the set of standard binary trees, and $\mathcal{T}_3(n)$ the set of standard ternary trees (see Fig. 1). It is well known that, for any $n \geq 0, d \geq 1$,

$$|\mathcal{T}_d(n)| = \binom{dn+1}{n}/(dn+1); \tag{1}$$

this can be proved, by decomposing d-ary trees at their root, and by a simple recurrence, or also by the rotation principle (see Sec. 2.5). Each tree $t \in \mathcal{T}_d(n)$ satisfies

$$|t| \;=\; dn+1, \quad |t^o| = n, \quad |\partial t| = (d-1)n+1, \quad |E(t)| = dn, \tag{2}$$

where $|t|$ denotes the number of nodes, t^o the set of internal nodes of t, ∂t the set of leaves of t, and $E(t)$ the set of edges of t.

The main aim of this paper is to describe a procedure which produces a random uniform tree $\mathbf{t}_{n+1}$ in $\mathcal{T}_d(n+1)$, when one possesses a uniform tree $\mathbf{t}_n$ of $\mathcal{T}_d(n)$ using some simple operations and the introduction of some (small) additional randomness, of course. The construction we propose, similar in spirit to Rémy's construction [10] (binary case), is different from it, even in the case $d = 2$ (we discuss the differences in Section 4 and provide a third construction in the case $d = 2$). By n application of our procedure, it is possible to sample a uniform element of $\mathcal{T}_d(n)$ starting from a uniform tree in $\mathcal{T}_d(0)$ (the tree reduced to its root). The details will be given in Section 3.

Related works. Additionally to Rémy's construction, Marchal [7] proved that Rémy's construction can be used to build an increasing sequence of uniform Dyck path (since uniform binary trees with n internal nodes can be encoded by uniform Dyck path with $2n$ steps), and proved that normalized by $\sqrt{n}$, this sequence of Dyck path converges almost surely in $C[0,1]$ equipped with the topology of uniform convergence. Evans, Grübel and Wakolbinger study the Doob-Martin boundary of Rémy's tree growth chain in [5].

Bettinelli [3], through a bijective approach, gives a method to construct uniform rooted quadrangulations with n faces inductively (from a uniform quadrangulations with $n-1$ faces), and a related method to build uniform forests (with a fixed number of edges).

Haas & Stephenson [6] study a model of growing d-ary trees, with a construction similar to that of Rémy's, that is, a node with degree d is inserted at each round "inside a uniformly chosen random edge"; but in the case of d-ary tree (with $d \geq 3$), as proved in [6], this method does not produce a sequence of uniform d-ary trees; even the order of the height of this model of trees does not fit with uniform d-ary tree, since the height order is $n^{1/k}$, when uniform d-ary trees have a height of order $\sqrt{n}$ (Aldous [1]).

Other methods exist to simulate uniform d-ary tree with n internal nodes: see e.g. Devroye [4] and Bacher [2].

From a bijection to a growing procedure. The construction we propose relies on a new bijection between a set of edge-marked trees with n internal nodes (built over $\mathcal{T}_d(n)$), and a set of leaf-marked trees with $n+1$ internal nodes (built over $\mathcal{T}_d(n+1)$). Let us start from the following simple observation: (1) is equivalent to

$$\binom{(d-1)(n+1)+1}{d-1} \times |\mathcal{T}_d(n+1)| = d \times \binom{dn+d-1}{d-1} \times |\mathcal{T}_d(n)|. \tag{3}$$

Let us interpret the different elements appearing in this relation. For S a set, and $m \geq 0$, denote by $\mathsf{Subset}_m(S)$ the set of subsets of S with cardinality m. Of course, $|\mathsf{Subset}_m(S)| = \binom{|S|}{m}$.

Definition 1. *For $n \geq 0$, denote by $\mathcal{T}_d^{\bullet,m}(n)$ the set of pairs $(\mathbf{t}, \boldsymbol{\ell})$ where $\mathbf{t} \in \mathcal{T}_d(n)$ and $\boldsymbol{\ell} \in \mathsf{Subset}_m(\partial\mathbf{t})$: in words the set of d-ary trees with n internal nodes and m distinguished leaves. An element of $\mathcal{T}_d^{\bullet,m}(n)$ is called a m-leaf-marked trees of size n.*

Since all trees in $\mathcal{T}_d(n+1)$ have $(d-1)(n+1)+1$ leaves, we have

$$|\mathcal{T}_d^{\bullet,d-1}(n+1)| \;=\; \binom{(d-1)(n+1)+1}{d-1} \times |\mathcal{T}_d(n+1)|, \tag{4}$$

and this is precisely the left hand side of (3).

Introduce a set of cardinality $d-1$, which can be seen as an ordered list of extra available edges:

$$\mathsf{Buds}(d) = \{b_0, \cdots, b_{d-2}\}. \tag{5}$$

Definition 2. *For any $n \geq 0$, we denote by $\mathcal{T}_d^{\blacksquare,d-1}(n)$ the set of pairs $(\mathbf{t}, \boldsymbol{e})$ where $\mathbf{t} \in \mathcal{T}_d(n)$ and $\boldsymbol{e}$ is an element of $\mathsf{Subset}_{d-1}(E(\mathbf{t}) \cup \mathsf{Buds}(d))$; this is the set of edges marked trees of size n. Hence, the marks are shared between $\mathsf{Buds}(d)$ and the edge set $E(\mathbf{t})$ of $\mathbf{t}$, and their total number is $d-1$.*

By (2), $|E(\mathbf{t}) \cup \mathsf{Buds}(d)| = dn+d-1$ for any tree on $\mathbf{t} \in \mathcal{T}_d(n)$, so that

$$|\mathcal{T}_d^{\blacksquare,d-1}(n)| = \binom{dn+d-1}{d-1} \times |\mathcal{T}_d(n)|, \tag{6}$$

and then the right hand side of (3) is:

$$|\mathcal{T}_d^{\bullet,d-1}(n) \times [\![1,d]\!]| \quad = \quad d \times \binom{dn+d-1}{d-1} \times |\mathcal{T}_d(n)|. \tag{7}$$

The main part of the rest of the paper is devoted to describe a new bijection between $\mathcal{T}_d^{\bullet,d-1}(n+1)$ and $\mathcal{T}_d^{\bullet,d-1}(n) \times [\![1,d]\!]$. It worth probably a moment thought if it is not clear enough: an explicit bijection between these sets allows to construct a procedure to produce a uniform tree $\mathbf{t}_{n+1}$ in $\mathcal{T}_d(n+1)$ from a uniform tree $\mathbf{t}_n$ in $\mathcal{T}_d(n)$ (see Section 3).

To describe our bijection between $\mathcal{T}_d^{\bullet,d-1}(n+1)$ and $\mathcal{T}_d^{\bullet,d-1}(n) \times [\![1,d]\!]$, we will need two additional families of objects: The set of leaf marked forests (see Fig. 2).

Definition 3. *We set*

$$\mathcal{F}_d^{\bullet,d-1}(n) = \bigcup_{\substack{n_1,\cdots,n_d \geq 0 \\ n_1+\cdots+n_d=n}} \bigcup_{\substack{m_1,\cdots,m_d \geq 0 \\ m_1+\cdots+m_d=d-1}} \mathcal{T}_d^{\bullet,m_1}(n_1) \times \cdots \times \mathcal{T}_d^{\bullet,m_d}(n_d), \tag{8}$$

that is the set of leaf-marked forests $f = ((f^{(0)},\ell^{(0)}),\cdots,(f^{(d-1)},\ell^{(d-1)}))$ made of d trees, each of them being a d-ary tree, with a total number of n internal nodes, and a total of $d-1$ marked leaves.

For such a leaf-marked forest f, define the "leaf sequence" $s(f) = (s_i(f), 0 \leq i \leq d)$, defined by

$$s_i(f) \quad = \quad (|\ell^{(0)}|-1) + \cdots + (|\ell^{(i-1)}|-1) \text{ for } i \in [\![0,d]\!] \tag{9}$$

which is the path starting at 0, and whose increments are the $|\ell^{(j)}|-1$. Since there are $d-1$ marks, we have $s_d(f) = -1$.

Denote by $\mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)$ the subset of $\mathcal{F}_d^{\bullet,d-1}(n)$ made of excursion type forests

$$\mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n) = \left\{ f \in \mathcal{F}_d^{\bullet,d-1}(n) \ : s_i(f) \text{ for all } i \in \{0,\cdots,d-1\}, s_d(f) = -1 \right\}. \tag{10}$$

In fact, to be an excursion type forest is a property concerning the leaf sequence. Since the map which return the list of subtrees rooted at the children of the root, is a bijection from $\mathcal{T}_d^{\bullet,d-1}(n+1)$ onto $\mathcal{F}_d^{\bullet,d-1}(n)$,

$$|\mathcal{F}_d^{\bullet,d-1}(n)| = \left|\mathcal{T}_d^{\bullet,d-1}(n+1)\right|. \tag{11}$$

We also have by the application of the rotation principle (see Section 2.5 for some recall) that

$$|\mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)| = |\mathcal{F}_d^{\bullet,d-1}(n)| \,/\, d. \tag{12}$$

We then have by (3), (4),(6) and (11)

$$\left|\mathcal{T}_d^{\bullet,d-1}(n) \times [\![1,d]\!]\right| = \left|\mathcal{F}_d^{\bullet,d-1}(n)\right| = \left|[\![1,d]\!] \times \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)\right| = \left|\mathcal{T}_d^{\bullet,d-1}(n+1)\right|. \tag{13}$$

All these sets having the same cardinality, there exists some bijective correspondences between them, but for the random generation purpose, we want to propose some bijections that preserves as much as

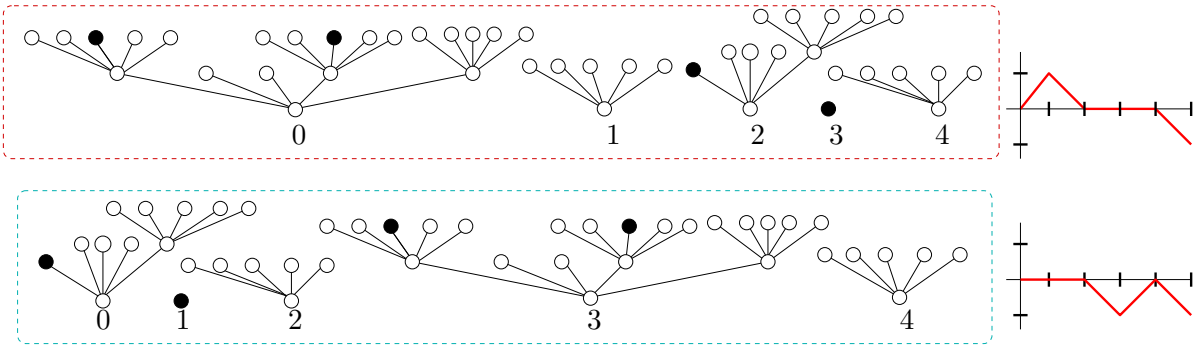

Figure 2: The first forest belongs to $\mathcal{F}_5^{\bullet,d-1}(8)$: it is a 5-ary forest with a total of 8 internal nodes and 4 marked leaves, and 5 roots. Since $(|\ell^{(0)}|,\cdots,|\ell^{(4)}|)=(2,0,1,1,0)$, the associated leaf sequence has increments $(1,-1,0,0,-1)$ as represented on the path at the right of the forest. The leaf sequence is non negative except at the very last position, so that this forest belongs to $\mathcal{F}_5^{\bullet,d-1,\mathsf{Ex}}(8)$. The second forest belongs to $\mathcal{F}_5^{\bullet,d-1}(8)$, but now $(|\ell^{(0)}|,\cdots,|\ell^{(4)}|)=(1,1,0,2,0)$, from what it can be seen that the associated leaf sequence is negative at position 3, so that this forest is not in $\mathcal{F}_5^{\bullet,d-1,\mathsf{Ex}}(8)$.

possible the forest/tree structures. We will decompose our bijection between $\mathcal{T}_d^{\bullet,d-1}(n)\times[\![1,d]\!]$ and $\mathcal{T}_d^{\bullet,d-1}(n+1)$ as the composition of three bijections,

$$\begin{array}{ccccccc}
 & \xrightarrow{\mathsf{Cut}} & & & \xrightarrow{\mathsf{Rotate}} & & \\
\mathcal{T}_d^{\bullet,d-1}(n)\times[\![1,d]\!] & \longmapsto & \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)\times[\![1,d]\!] & & \longmapsto & \mathcal{F}_d^{\bullet,d-1}(n) & \\
((\mathbf{t}_n,\mathbf{e}),\mathbf{a}) & \longmapsto & \left[\left(\left(\mathbf{g}^{(i)},\boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right),0\le i\le d-1\right),\mathbf{a}\right] & \longmapsto & \left(\left(\mathbf{f}^{(i)},\boldsymbol{\ell}^{(i)}\right),0\le i\le d-1\right) & & \\
 & \xrightarrow{\mathsf{AddRoot}} & & & & & \\
 & \longmapsto & \mathcal{T}_d^{\bullet,d-1}(n+1) & & & & \\
 & \longmapsto & (\mathbf{t}_{n+1},\boldsymbol{\ell}) & & & &
\end{array}$$

NB: we should have marked a dependence in (n,d) in these bijections (with an index or an exponent), but renounce to do it, to avoid too heavy notations.

In fact, we will see that

Lemma 4. *Each of the map* Cut, Rotate, AddRoot *is a bijection so that*

$$\mathsf{Enlarge}:\mathcal{T}_d^{\bullet,d-1}(n)\times[\![1,d]\!]\mapsto\mathcal{T}_d^{\bullet,d-1}(n+1) \tag{14}$$

defined by $\mathsf{Enlarge}:=\mathsf{AddRoot}\circ\mathsf{Rotate}\circ\mathsf{Cut}$ *is a bijection, whose inverse is*

$$\mathsf{Reduce}:=\mathsf{Cut}^{-1}\circ\mathsf{Rotate}^{-1}\circ\mathsf{AddRoot}^{-1}. \tag{15}$$

Figure 3 allows to understand the mechanism of the bijection, and it is even possibly sufficient to some readers to understand the complete picture.

2 Formal description of the bijections and proofs

2.1 Classical encoding of trees

To define formally the trees, we use the language of set theory (as proposed by Neveu [8]). See Fig. 4 for an illustration. Set $\mathsf{Alphabet}(d)=[\![1,d]\!]$, and the associated set of finite words is

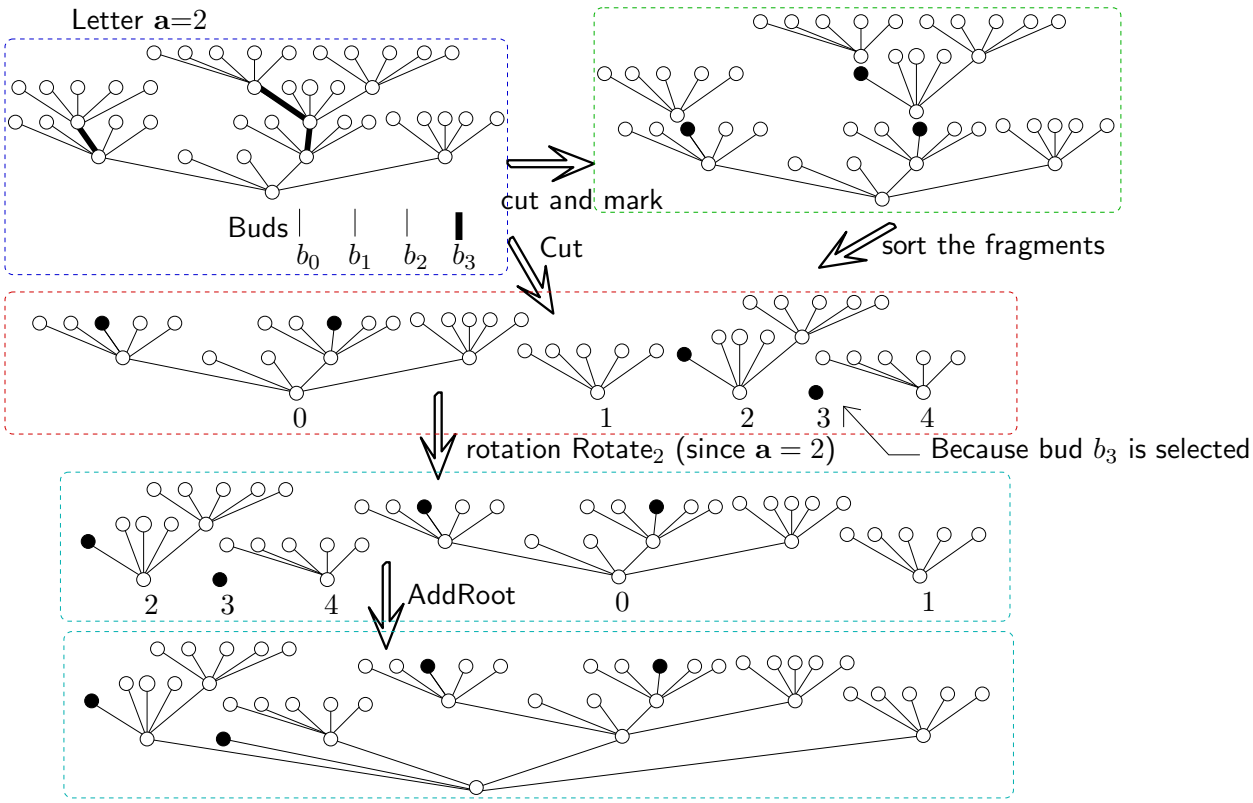

Figure 3: Illustration of the Cut, Rotate, AddRoot maps applied to an element $((\mathbf{t},\mathbf{e}),\mathbf{a})$ with $\mathbf{a}=2$ of $\mathcal{T}_5^{\blacksquare,d-1}(8)\times\{1,\cdots,5\}$, in which the buds b_2 is marked as well as 3 edges of $\mathbf{t}$ (darkened).

• Cut disconnects the tree at the top extremity of all marked edges, remove the marks on the edges, marks the top extremities of the ancient marked edges, and sort the fragments according to their least vertices for the lexicographical order, in the initial tree. If the buds $(b_{i_j}, 1\le j\le m)$ are selected, then some trees reduced to marked nodes are inserted in the fragment list, so that that these marked nodes represent the trees $\mathbf{t}_{i_j}$ in the final forest (here b_3 is selected so the tree t_3 in the list (t_0,t_1,t_2,t_3,t_4) is a simple marked node). This gives an element of $\mathcal{F}_5^{\bullet,d-1,\mathsf{Ex}}(8)$ (since the corresponding walk $s(f)$ satisfies $s(f)=(0,1,0,0,0,-1)$).

• Rotate is simple enough since it shifts the indices of the forest by $\mathbf{a}$ in $\mathbb{Z}/d\mathbb{Z}$, where $\mathbf{a}$ is the selected letter. It produces an element of $\mathcal{F}_5^{\bullet,d-1}(8)$ (here since $\mathbf{a}=2$, the image is (t_2,t_3,t_4,t_0,t_1)). The fact that it is invertible is not evident since $\mathbf{a}$ must be recovered from the image only.

• AddRoot is totally trivial: its name describes its action.

Finally, after composition of the three maps, and element of $\mathcal{T}_5^{\bullet,4}(9)$ (9 internal nodes, 4 marked leaves) is produced.

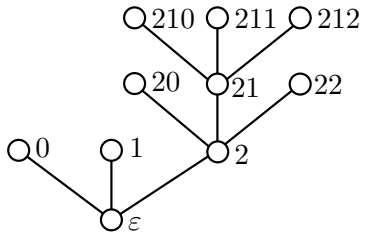

Figure 4: This tree is $\{\varepsilon,0,1,2,20,21,22,210,211,212\}$.

$$\mathsf{Words}(d) = \{\varepsilon\} \cup \bigcup_{m\geq 1} \mathsf{Alphabet}(d)^m,$$

where ε is the empty word. The set $\mathsf{Words}(d)$ is the infinite complete d-ary tree: each word w stands for a node, and wj stands for the jth child of w (see Figure 4). The vocabulary of genealogy is introduced: if u is a prefix of v, then u is called an ancestor of v, the largest strict prefix of v is the parent $p(v)$ of v; the node ε is the root.

A d-ary tree is a subset T of $\mathsf{Words}(d)$ satisfying the following constraints:
• T contains ε,
• T is stable by prefix ($(u_1, \cdots, u_h) \in T$ implies that $(u_1, \cdots, u_{h-1}) \in T$),
• T contains the d descendants of each internal node: $(u_1, \cdots, u_h) \in T$ implies that $(u_1, \cdots, u_{h-1}, j) \in T$ for all $j \in \mathsf{Alphabet}(d)$.

An edge of T is a pair $(u, p(u))$ between a node u of T and its parent $p(u)$, assumed to be directed toward $p(u)$. The lexicographical order makes of each tree a totally ordered set: the prefix of a word is smaller than the word itself, and ε is the smallest element of $\mathsf{Words}(d)$.

2.2 The map Cut

Let us describe the map

$$\begin{array}{rccl} \mathsf{Cut}: & \mathcal{T}_d^{\blacksquare,d-1}(n) \times [\![1,d]\!] & \longrightarrow & \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n) \times [\![1,d]\!] \\ & ((\mathbf{t}_n, \mathbf{e}), \mathbf{a}) & \longmapsto & \left[\left(\left(\mathbf{g}^{(i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right), 0 \leq i \leq d-1\right), \mathbf{a}\right] \end{array} \tag{16}$$

Observe that the letter $\mathbf{a}$ is not modified, and in fact, it is even not used to define Cut. It is present only because we are dealing with some chained bijections, and it is needed to define Rotate. In this section, it can be ignored.

■ Let us concentrate only the definition of $(\mathbf{t}_n, \mathbf{e}) \to \left(\left(\mathbf{g}^{(i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right), 0 \leq i \leq d-1\right)$.

The set $\mathbf{e}$ contains $d-1$ elements: $k(1)$ are edges of $\mathbf{t}_n$ and $k(2)$ are buds $(b_{n_1}, \cdots, b_{n_{k(2)}})$ for some $k(1)+k(2) = d-1$. The buds are sorted according to their indices $0 \leq n_1 < n_2 < \cdots < n_{k(2)} \leq d-2$.

We decompose the bijection in 2 steps. The two first lines of Fig. 3 illustrate the construction.

Step 1: for any $i \in \{1, \cdots, k(2)\}$, the selected bud b_{n_i} is transformed into the tree $\left(\mathbf{g}^{(n_i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(n_i)}\right)$, of the image. This tree is reduced to its root $\mathbf{g}^{(n_i)} = \{\varepsilon\}$, and it is marked at it, so that $\boldsymbol{\ell}_{\mathbf{g}}^{(n_i)} = \{\varepsilon\}$. This produces $k(2)$ trees.

Step 2: Construct the increasing sequence $\mathsf{RemainingIndices}$ of the trees $\left(\mathbf{g}^{(j)}, \boldsymbol{\ell}_{\mathbf{g}}^{(j)}\right)$ that remains to be built; it is made by the $d-k(2) = k(1)+1$ elements of $\{0, \cdots, d-1\} \setminus \{n_1, \cdots, n_{k(2}\}$ sorted increasingly.

The construction of these marked trees is done by an algorithm. Some marked edges are progressively transformed into marked leaves, so that the current tree on which we are working, which is an edge marked tree at the beginning, may become temporally a tree marked at some edges and some leaves: **an edge-and-leaf-marked tree** is a 3-tuple (t, e, ℓ) where $t \in \cup_n \mathcal{T}_d(n)$, the marked edges $e \subset (E(t) \cup \mathsf{Buds}(d))$, the marked leaves $\ell \subset \partial t$.

At the beginning, set some variables: $(t, e, \ell) := (\mathbf{t}_n, \mathbf{e}, \varnothing), \mathsf{Rem} = \mathsf{RemainingIndices}$. These variables will evolve during the algorithm execution.

Initially $|\mathsf{Rem}| = |e| + 1$, and this property will be preserved all along the algorithm execution ($|\mathsf{Rem}|$ and $|e|$ decrease simultaneously).

Repeat until the complete disappearance of the elements of e

■ Take $i^\star := \max\{\mathsf{Rem}\}$ the largest remaining index to treat.

■ If e is empty, then there are no marked edge, so that Rem contains only $i^\star$. Set $\left(\mathbf{g}^{(i^\star)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i^\star)}\right) = (t, \ell)$ and the algorithm finishes here.

■ If e is not empty, then find $(u, p(u))$ the largest marked edge of e for the lexicographical order.

– First intuitively, the marked tree $\left(\mathbf{g}^{(i^\star)}, \boldsymbol{\ell}^{(i^\star)}\right)$ is the subtree of t attached at u. Formally, we need to express this using our formalism, a tree being a set of words (containing ε). Define the set of descendants of u in t is

$$\mathsf{Descendants}_t(u) := \{w \in t \ : u \text{ is a prefix of } w\}$$

(including u): the marked leaves supported by this set as

$$\ell(u) := \ell \cap \mathsf{Descendants}_t(u).$$

We have now to remove the prefix u to all the elements of $\mathsf{Descendants}_t(u)$ to turn it into a tree (in a tree, the root is the empty word ε, when in the set $\mathsf{Descendants}(u)$, the natural root is u). Set

$$\left\{ \begin{array}{rcl} \mathbf{g}^{(i^\star)} & = & \{w \in \mathsf{Words}(d), uw \in \mathsf{Descendants}_t(u)\}, \\ \boldsymbol{\ell}^{(i^\star)} & = & \{w \in \mathsf{Words}(d), uw \in \ell(u)\}. \end{array} \right. \tag{17}$$

– Now, detach from t the subtree attached at u (see Fig. 3), and update everything as detailed in the four following points:
• set $t := (t \setminus \mathsf{Descendants}(u)) \cup \{u\}$ the node u is somehow duplicated, and the descendants of u removed from t
• $\ell = (\ell \setminus \ell(u)) \cup \{u\}$ meaning that u, which is now a leaf of t, is marked and then, added to ℓ, when the marked leaf of the descendant of u are removed, if any
• Remove $(u, p(u))$ from the set of marked edges e (set $e = e \setminus \{(u, p(u))\}$),
• set $\mathsf{Rem} := \mathsf{Rem} \setminus \{i^\star\}$ (the tree $(\mathbf{g}^{(i^\star)}, \boldsymbol{\ell}^{(i^\star)})$ is fixed, so that the index $i^\star$ is removed from the set of indices to be treated).

——— End of the algorithm ———

Lemma 5. Cut *is a bijection.*

Proof. First, let us see why Cut is injective: probably the first point to notice is that the trees created by Step 2 can also be reduced to their root, but in this case, this root is not marked. Taking this into account, the fact that the images of different elements in $\mathcal{T}_d^{\blacksquare,d-1}(n) \times [\![1, d]\!]$ are different is simple to see. What is less simple, is the fact that the image of Cut is indeed included in $\mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n) \times [\![1, d]\!]$, that is $\left((\mathbf{g}^{(i)}, \boldsymbol{\ell}^{(i)}), 1 \le i \le d\right) \in \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)$. In view of (12) it would be enough to conclude.

As a matter of fact, it is simple to see that the image of each edge marked tree from $\mathcal{T}_d^{\blacksquare,d-1}(n)$ is a forest marked at $d-1$ leaves (each marked bud and marked edge is, at some time, transformed into a marked leaf). Only the fact that this forest has the excursion type needs to be shown.

The indices $n_1, \cdots, n_{k(2)}$ of the buds $b_{n_1}, ..., b_{n_{k(2)}}$ becomes the indices of some trees reduced to their roots, marked. Hence, the corresponding increments $|\ell^{(n_i)}|-1$ equal 0. Hence, the fact that the excursion property is satisfied depends basically on the other increments.
Notice that the index $n_{k(2)} < d-1$ since the buds are labeled from 0 to $d-2$: the last tree of the forest $(\mathbf{g}^{(d-1)}, \boldsymbol{\ell}^{(d-1)})$ comes from the first detached fragment of t, so that it has no marked leaf, and thus, the last increment is $|\ell^{(d-1)}| - 1 = -1$ (as for all forests of excursion type).

It remains to prove that $s_j(f) = \sum_{k=0}^{j-1} |\ell^{(k)}|-1 \geq 0$ for $j \leq d-1$. Since, the buds contribution $|\ell^{(n_i)}|-1$ equal 0, somehow, we can ignore these increments and restrict ourselves to the trees $((\mathbf{g}^{(m_j)}, \boldsymbol{\ell}^{(m_j)}), 1 \leq j \in K)$ where $m_1, \cdots, m_K$ is the list of indices (taken under their initial order) of the trees obtained by the decomposition of t (using Step 2). Now, the conclusion is simple: each tree $\mathbf{g}^{(m_i)}$ upon creation, creates a leaf in another other component with smaller index (since it is still attached to the current tree t). Hence, for any ℓ,

$$N_j := |\ell^{(m_1)}| + \sum_{k=2}^{j-1} (|\ell^{(m_k)}| - 1) = s_{m_j}(f) + 1$$

can be interpreted as the number of fragments that were attached to the leaves of the union of the fragments $m_1, \cdots, m_{j-1}$ from what we removed 1, for the fragments 2 to $j-1$, so that N_j is the number of fragments that were attached to the $j-1$ first fragments, and which have not been visited yet: it is clear that $N_j \geq 1$, as long as $j < K-1$ from what the conclusion follows. □

The map Cut^{-1}

We present the bijection Cut^{-1} which is simpler:

$$\begin{array}{rccc} \mathsf{Cut}^{-1}: & \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n) \times [\![1,d]\!] & \longrightarrow & \mathcal{T}_d^{\blacksquare,d-1}(n) \times [\![1,d]\!] \\ & \left[\left(\left(\mathbf{g}^{(i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right), 0 \leq i \leq d-1\right), \mathbf{a}\right] & \longmapsto & ((\mathbf{t}_n, \mathbf{e}), \mathbf{a}) \end{array} \tag{18}$$

Again, the letter $\mathbf{a}$ is preserved so that let us focus on the rest. Assume that $\left(\left(\mathbf{g}^{(i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right), 0 \leq i \leq d-1\right)$ is given, and let us "reconstruct" $(\mathbf{t}_n, \mathbf{e})$.
Step 1. Construct the set S of indices i such that $\mathbf{g}^{(i)} = \{\varepsilon\}$ and $\boldsymbol{\ell}_{\mathbf{g}}^{(i)} = \{\varepsilon\}$, which corresponds to trees reduced to a single node, which is marked. From what is said above, for all such $i \in S$, $i < d-1$. The subset of returned marked buds will be $\{b_i, i \in S\}$: set temporarily $\mathbf{e} = \{b_i, i \in S\}$, since it is part of the set $\mathbf{e}$ to be defined.
Step 2. For the sequence of $\mathsf{RemainingIndices} = (m_1, \cdots, m_{d-|S|})$ sorted increasingly. Set

$$(T, L, E) = (\mathbf{g}^{(m_1)}, \boldsymbol{\ell}^{(m_1)}, \varnothing)$$

a variable of the algorithm, which is an edge and leaf marked tree, which at the beginning, coincides with the right most element of the list of remaining trees to be treated (those that are not reduced to a marked leaf). It will evolve during the algorithm execution; at the beginning it is not marked on any edges.

For $k = 2$ to $d - |S|$ do the following:
• Plug the tree $\mathbf{g}^{(m_k)}$ at the first leaf u (for the lex. order) of T.
• Add the edge $(u, p(u))$ at E (set $E = E \cup \{(u, p(u))\}$). Formally, the tree T obtained after this operation

is $T = T + u\mathbf{g}^{(m_k)}$, and $L = (L \cup u\boldsymbol{\ell}^{(m_k)}) \setminus \{u\}$ (since the prefix u, added to $\mathbf{g}^{(m_k)}$ gives a subtree of T isomorphic to $\mathbf{g}^{(m_k)}$).

At the end set $(\mathbf{t}_n, \mathbf{e}) = (T, E)$.

The fact that the function Cut^{-1} is indeed the inverse map of Cut^{-1} should be clear, and is left as an exercise.

2.3 The map Rotate

The rotate map is illustrated on Fig. 3

$$\begin{array}{rccc} \mathsf{Rotate}: & \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n) \times [\![1,d]\!] & \longrightarrow & \mathcal{F}_d^{\bullet,d-1}(n) \\ & \left[\left(\left(\mathbf{g}^{(i)}, \boldsymbol{\ell}_{\mathbf{g}}^{(i)}\right), 0 \le i \le d-1\right), \mathbf{a}\right] & \longmapsto & \left(\left(\mathbf{f}^{(i)}, \boldsymbol{\ell}^{(i)}\right), 0 \le i \le d-1\right) \end{array} . \tag{19}$$

This map is just the rotation of indices (by $\mathbf{a}$ in $\mathbb{Z}/d\mathbb{Z}$) defined by

$$\left(\mathbf{f}^{(i)}, \boldsymbol{\ell}^{(i)}\right) := \left(\mathbf{g}^{(i+\mathbf{a} \mod d)}, \boldsymbol{\ell}_g^{(i+\mathbf{a} \mod d)}\right), \quad \text{for all } i \in \mathbb{Z}/d\mathbb{Z}.$$

The fact that this map is invertible is one of the key point of the proof: it is not that obvious because, $\mathbf{a}$ needs to be recovered too! The argument is developed in Section 2.5 (second statement of Lemma 6). Using this Lemma, given an element $F := \left(\left(\mathbf{f}^{(i)}, \boldsymbol{\ell}^{(i)}\right), 0 \le i \le d-1\right) \in \mathcal{F}_d^{\bullet,d-1}(n)$, there exists a unique element $\mathbf{b} \in [\![1,d]\!]$ such that $\left(\left(\mathbf{f}^{(i+\mathbf{b} \mod d)}, \boldsymbol{\ell}^{(i+\mathbf{b} \mod d)}\right), 0 \le i \le d-1\right) \in \mathcal{F}_d^{\bullet,d-1,\mathsf{Ex}}(n)$, and then

$$\mathsf{Rotate}^{-1}\left(\mathbf{f}^{(i)}, \boldsymbol{\ell}^{(i)}\right) = \left[\left(\left(\mathbf{f}^{(i+\mathbf{b} \mod d)}, \boldsymbol{\ell}^{(i+\mathbf{b} \mod d)}\right), 0 \le i \le d-1\right), \mathbf{b}\right].$$

2.4 The map AddRoot

The map AddRoot is totally trivial and its name suffices to understand its action.

$$\begin{array}{rccc} \mathsf{AddRoot}: & \mathcal{F}_d^{\bullet,d-1}(n) & \longrightarrow & \mathcal{T}_d^{\bullet,d-1}(n+1) \\ & \left(\left(\mathbf{f}^{(i)}, \boldsymbol{\ell}^{(i)}\right), 0 \le i \le d-1\right) & \longmapsto & (\mathbf{t}_{n+1}, \boldsymbol{\ell}) \end{array}$$

Formally, we have

$$\mathbf{t}_{n+1} = \{\varepsilon\} \bigcup_{i=0}^{d-1} i\mathbf{f}^{(i)}, \qquad \boldsymbol{\ell} = \bigcup_{i=0}^{d-1} i\boldsymbol{\ell}^{(i)},$$

in words: $\mathbf{t}_{n+1}$ is the marked tree whose subtrees form the original forest[1].

■ The reverse map $\begin{array}{rccc} \mathsf{AddRoot}^{-1}: & \mathcal{T}_d^{\bullet,d-1}(n+1) & \longrightarrow & \mathcal{F}_d^{\bullet,d-1}(n) \\ & (\mathbf{t}_{n+1}, \boldsymbol{\ell}) & \longmapsto & \left(\left(\mathbf{f}^{(0)}, \boldsymbol{\ell}^{(0)}\right), \cdots, \left(\mathbf{f}^{(d-1)}, \boldsymbol{\ell}^{(d-1)}\right)\right) \end{array}$, amounts to removing the root of $\mathbf{t}_{n+1}$ while conserving the marked leaves, and to returning the sequence of subtrees of $\mathbf{t}_{n+1}$ rooted at the children of the root, according to their initial order.

[1] again $i\mathbf{f}^{(i)}$ is the set by adding i as a prefix to all the nodes of $\mathbf{f}^{(i)}$, so that in $\mathbf{t}_{n+1}$ the subtree rooted at i is isomorphic to $\mathbf{f}^{(i)}$

2.5 The rotation principle

Set

$$\begin{aligned} \mathsf{Seq}_m &= \{s[\![0,m]\!] \ : s_0 = 0, s_m = -1, s_{j+1} - s_j \geq -1, \ \forall\ 0 \leq j \leq m-1\}, \qquad (20) \\ \mathsf{Excursions}(m) &= \mathsf{Seq}_m \cap \{s[\![0,m]\!] \ : s_0 \geq 0, \cdots, s_{m-1} \geq 0, s_m = -1\}. \qquad (21) \end{aligned}$$

The first set is sometimes called the set of Łukasiewicz walks, and the second, the set of excursions. They are set of length m paths with integer values, and increment bounded from below by -1, that ends at -1; excursions have the additional property to hit -1 for the first time at the end.

Denote by $\Delta s_{j-1} = s_j - s_{j-1}$ the jth increment of the path s. Of course, the increments $(\Delta s_j, 0 \leq j \leq m-1)$ characterizes $s[\![0,m]\!]$, since

$$s_j = \Delta s_0 + \cdots + \Delta s_{j-1}.$$

For any $r \in [\![0, m-1]\!]$, the rth rotation is a map on Seq_m

$$\begin{array}{rccc} \mathsf{Rot}_r : & \mathsf{Seq}_m & \longrightarrow & \mathsf{Seq}_m \\ & s[\![0,m]\!] & \longmapsto & s'[\![0,m]\!] \end{array}$$

which is better seen on the increments, which are subjected to a simple rotation around the $\mathbb{Z}/m\mathbb{Z}$:

$$\Delta s'_i = \Delta s_{i+r \mod m}, \quad \text{for } i \in [\![0, m-1]\!].$$

A rotation class of an element $s \in \mathsf{Seq}_m$ is defined by

$$\mathsf{RotationClass}(s) = \{\mathsf{Rot}_0(s), \mathsf{Rot}_1(s), \cdots, \mathsf{Rot}_{m-1}(s)\},$$

and two elements s and s' of Seq_m are said to be in the same rotation class, if there exists $r \in [\![0, m-1]\!]$ such that $s = \mathsf{Rot}_r(s')$. Of course $G_m := \{\mathsf{Rot}_r, 0 \leq r \leq m-1\}$ is a group for the composition, isomorphic to $(\mathbb{Z}/m\mathbb{Z}, +)$, so that, it is easily seen that rotation classes are equivalence classes.

The following result can be found under several forms in the combinatorics literature (rotation principle), and can be found in Otter [9]:

Lemma 6. *For any $m \geq 1$, each rotation class possesses m different elements, and exactly one of these element belongs to* $\mathsf{Excursions}(m)$.
Hence, for any $s \in \mathsf{Seq}_m$, there exists a unique $a \in \mathbb{Z}/m\mathbb{Z}$ such that $\mathsf{Rot}_a(s) \in \mathsf{Excursions}(m)$.

Proof. Maybe, the simplest proof relies on the action of Rot_r on the first time $\min \operatorname{argmin}(s)$ a path s hits its minimum for the first time. Observe Fig. 5. Consider the rotation class of a given element $s \in \mathsf{Seq}(m)$. A simple analysis shows that, for $a(s) = \min \operatorname{argmin}(s)$, we have

$$\mathsf{Rot}_{a(s)}(s) \in \mathsf{Excursions}(m)$$

so that there is (at least) one excursion in each rotation class, and for any $s \in \mathsf{Excursions}(m)$

$$a(\mathsf{Rot}_r(s)) = m - r,$$

implying that all the elements of the rotation class of an excursion reaches its minimum for the first time at a different place, so that, all of them are different.

□

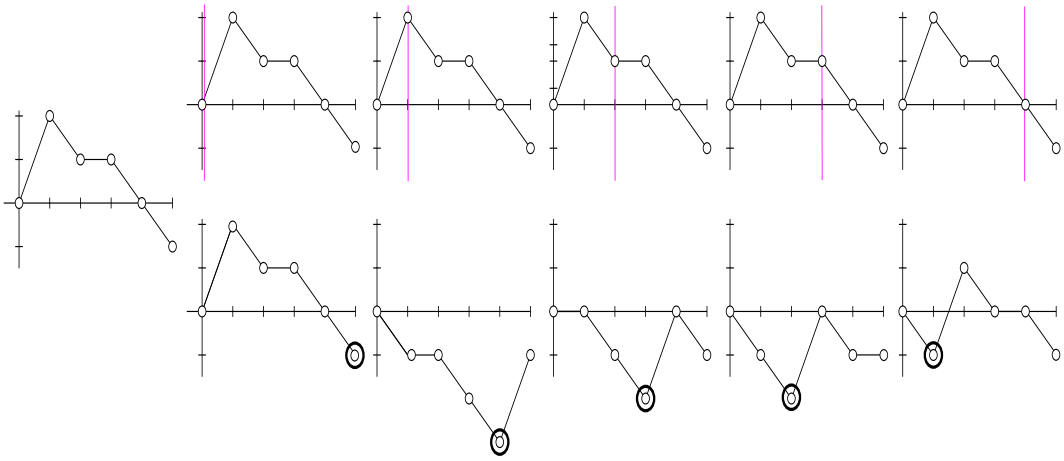

Figure 5: A path s from $\mathsf{Excursions}(m)$, for $m = 5$, and its 5 rotations $\mathsf{Rot}_0(s), \cdots, \mathsf{Rot}_{m-1}(s)$ on the line below: each of them are different (they reach their minimum for the first time at different places).

3 Random generation of a sequence of growing uniform trees

We first present the principle of the random generation, and then we will see why the sequence of bijections $(\mathsf{Enlarge}_{k,d}, k \geq 0)$ can be used to generate a uniform tree $\mathbf{t}_n$ in $\mathcal{T}_d(n)$ at a linear cost (for a cost model we will detail).

Assume that $\mathbf{t}_k$ is uniform in $\mathcal{T}_d(k)$ (start this recursion at $\mathbf{t}_0$ the tree reduced to its root ε).

——— Algo ———
choose a uniform subset $\mathbf{e}$ of $E(\mathbf{t}_k) \cup \mathsf{Buds}(d)$,
2. independently, choose a uniform letter $\mathbf{a} \in \mathsf{Alphabet}(d)$.
3. Compute $(\mathbf{t}_{k+1}, \boldsymbol{\ell}) := \mathsf{Enlarge}_{k,d}(\mathbf{t}_k, \mathbf{e}, \mathbf{a})$.
———————

When $(\mathbf{t}_{k+1}, \boldsymbol{\ell})$ has been computed, the tree $\mathbf{t}_{k+1}$ is obtained by a simple projection (which amounts to forgetting the marked edges).

Lemma 7. $\mathbf{t}_{k+1}$ *is uniform in* $\mathcal{T}_d(k+1)$

Proof. Each element in the support of $(\mathbf{t}_k, \mathbf{e}, \mathbf{a})$ has weight $1/(d \times |\mathcal{T}_d^{\blacksquare,d-1}(k)|)$, and the number of pairs $(\mathbf{t}_{k+1}, \ell)$ corresponding to a given $\mathbf{t}_{k+1} = t$ is the same for all trees t, that is the number of ways to mark the leaves of t, that is $\binom{(d-1)(n+1)+1}{d-1}$: hence by (6) and (3)

$$\mathbb{P}(\mathbf{t}_{k+1} = t) = \frac{\binom{(d-1)(k+1)+1}{d-1}}{d \times |\mathcal{T}_d^{\blacksquare,d-1}(k)|} = \frac{1}{\mathcal{T}_d(k+1)}.$$

□

About the cost of this generation algorithm

To define the cost we need to explain a bit how the map $\mathsf{Enlarge}$ can be programmed, how to proceed to make the number of elementary operations as small as possible, and to fix the cost of the elementary operations.

First, Enlarge needs to move typically $d-1$ subtrees of $\mathcal{T}_d(n)$. Hence, some efficient operations are needed to find and move these subtrees. If the tree is encoded using some pointers with a link between each node and its parent in the tree, we may assume that the addition of a new node has a constant cost[2], and the redirection of some links can be also supposed to have a constant cost, if a table with the list of the nodes is available. The choice of d different edges can be done simply by choosing random nodes, and by identifying each edge with the higher node it contains (to do that, pick some ranks $\lfloor (U(dn+d-1)) \rfloor$ using uniform random variable $U \sim \mathsf{Uniform}([0,1])$ till $d-1$ different ranks are obtained); this costs a $O(1)$ number of calls to the random number generator[3].

To get an efficient encoding of the generation algorithm, it is also needed to reach each existing node (those with ranks taken at random, notably) in a fast way. We assume that the cost to reach a given vertex has a constant time [4].

For this model, the total cost of this algorithm forms to produce a uniform tree in $\mathcal{T}_d(n)$ from the single tree of size 0, is linear in n.

4 Comparison with Rémy's bijection

This section is devoted to a small discussion of the difference between our algorithm in the case $d=2$ with Rémy's (they are indeed rather different). A third algorithm is presented in section 4.3. We skip some details and just talk about the bijection between $|\mathcal{T}_d^{\bullet,d-1}(n) \times \{1,2\}|$ and $\mathcal{T}_2^{\bullet,1}(n+1)$: trees of size n marked on a edge by 0 or 1 and trees of size $n+1$ marked on a leaf.

4.1 Rémy algorithm

Rémy's algorithm is illustrated on Fig. 6.

Start with a tree t with n internal nodes, and then, $2n$ edges. There is a marked edge which is in

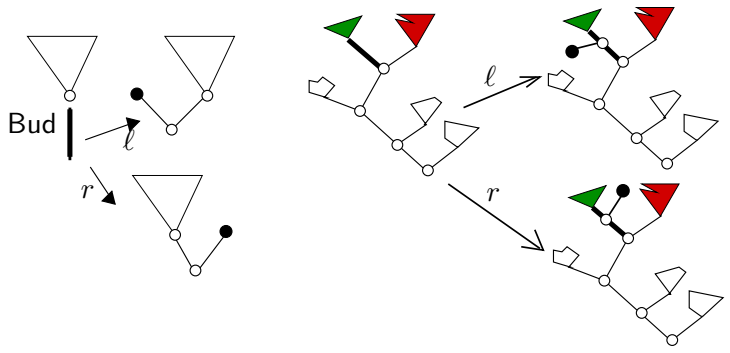

Figure 6: Illustration of Rémy algorithm

$E(t) \cup \mathsf{Buds}(2)$ with $\mathsf{Buds}(2) = \{b_0\}$ which is an available bud. There is a letter $\mathbf{a}$ in $\mathsf{Alphabet}(2) = \{1,2\}$, but in general, in the usual presentation of the bijection, the letter $\mathbf{a}$ is rather chosen in $\{r, \ell\}$, "right" or "left".

[2] it could be also natural to assume that this cost in $O(\log n)$, to take into account the size of the pointers

[3] an extra cost of $O(\log n)$ to take into account the bit cost of this random generation is also a natural model

[4] a cost $O(\log n)$ is also a natural model

The bijection on which Rémy's algorithm is built is

$$\begin{array}{rccc} R: & \mathcal{T}_d^{\blacksquare,d-1}(n) \times \{r,\ell\} & \longrightarrow & \mathcal{T}_2^{\bullet,1}(n+1) \\ & (t,e,a) & \longmapsto & (t',f) \end{array}$$

where:
■ if the selected edge $e = (u, p(u))$ is in $E(t)$ then do the following: insert a node w "at the middle" of the edge e, so that u is a child of w.
. If $\mathbf{a} = r$, then creates a right child of f and marks this leaf (and then u is the left child of w),
. If $\mathbf{a} = \ell$, then creates a left child of f and marks this leaf (and then u is the right child of w),
This gives a tree (t', f).
■ if the selected edge is the bud b_0, then add a new node u which will become the new root. To build t':
– if $\mathbf{a} = r$, add a right edge to u and mark the new leaf f at its extremity. Add a left edge from u to the root of t,
– if $\mathbf{a} = \ell$, add a left edge to u and mark the new leaf f at its extremity. Add a right edge from u to the root of t.

4.2 The new bijection in the binary case

The new bijection Enlarge presented in Section 1 is different from Rémy's bijection because it modifies dramatically the neighborhood of the selected edge $(u, p(u))$, by moving the subtree rooted at u at distance 1 of the root: see Figure 7.

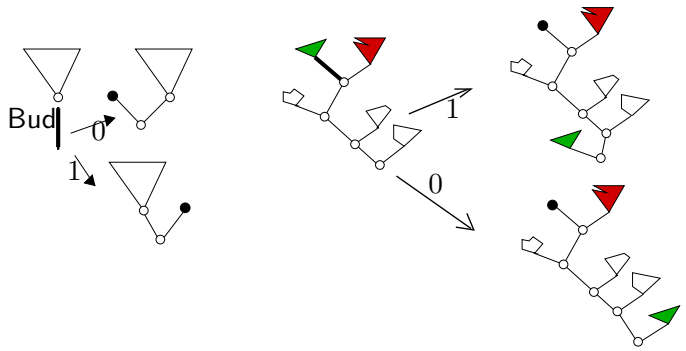

Figure 7: Illustration of the new algorithm: the tree at the end of the marked edge becomes a subtree of the root.

4.3 A third bijection in the binary case

We present a third bijection different from enlarge, and different from Rémy's, valid in the binary case too. It is illustrated on Fig. 8.
■ If the marked edge **e** is the bud, then do the same thing as in Rémy's bijection,
■ If the marked edge **e** is an edge $(u, p(u))$ of $E(t)$, then do the following: detach the subtree T grafted at u and mark the node u. Insert a new node v in the edge $(p(u), p(p(u)))$, add a node w as new child of v (as a right child if $\mathbf{a} = r$, on the left if $\mathbf{a} = \ell$), and graft the subtree T at w. If $p(u)$ is the root of the tree, then $p(p(u))$ is created too, as a new root.

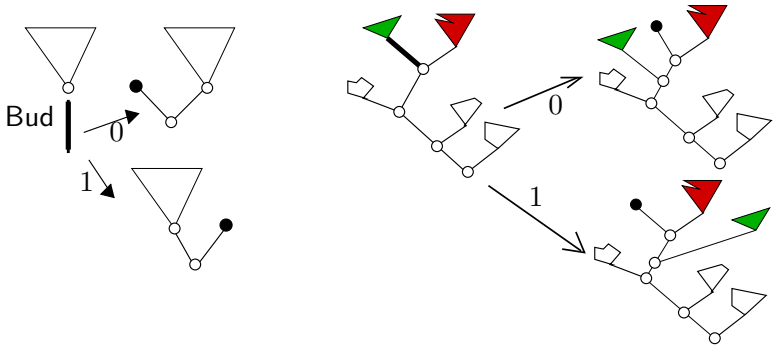

Figure 8: Illustration of the third algorithm

References

[1] D. Aldous. The continuum random tree. II. An overview. In Stochastic analysis (Durham, 1990), volume 167 of London Math. Soc. Lecture Note Ser., pages 23–70. Cambridge Univ. Press, Cambridge, 1991.

[2] A. Bacher. A new bijection on m-Dyck paths with application to random sampling. Arxiv:1603.06290v2.

[3] J. Bettinelli. Increasing forests and quadrangulations via a bijective approach. JCTA, 122:107–125, 2014.

[4] L. Devroye. Simulating Size-constrained Galton-Watson Trees. SIAM Journal on Computing, 2012, Vol.41, No.1.

[5] S. N. Evans, R. Grübel, and A. Wakolbinger. Doob–Martin boundary of Rémy's tree growth chain. The Annals of Probability, 45(1):225 – 277, 2017.

[6] B. Haas and R. Stephenson. Scaling limits of k-ary growing trees. ANN I H Poincare-PR, 51(4):1314 – 1341, 2015.

[7] P. Marchal. Constructing a sequence of random walks strongly converging to Brownian motion.

[8] J. Neveu. Arbres et processus de galton-watson. ANN I H Poincare-PR, 22(2):199–207, 1986.

[9] R. Otter. The Multiplicative Process. The Annals of Mathematical Statistics, 20(2):206 – 224, 1949.

[10] J.-L. Rémy. Un procédé itératif de dénombrement d'arbres binaires et son application à leur génération aléatoire. RAIRO Inform. Théor., 19(2):179–195, 1985.