Random-bit optimal uniform sampling for rooted planar trees with given sequence of degrees and Applications

O.Bodini [*], J. David [**], and Ph. Marchal

Institut Galilée, Université Paris 13, Villetaneuse (France)

Abstract. In this paper, we redesign and simplify an algorithm due to Remy et al. for the generation of rooted planar trees that satisfies a given partition of degrees. This new version is now optimal in terms of random bit complexity, up to a multiplicative constant. We then apply a natural process "simulate-guess-and-proof" to analyze the height of a random Motzkin in function of its frequency of unary nodes. When the number of unary nodes dominate, we prove some unconventional height phenomena (i.e. outside the universal $\Theta(\sqrt{n})$ behaviour.)

1 Introduction

Trees are probably among the most studied objects in combinatorics, computer science and probability. The literature on the subject is abundant and covers many aspects (analysis of structural properties such as height, profile, path length, number of patterns, but also dynamic aspects such as Galton-Watson processes or random generation, ...) and use various techniques such as analytic combinatorics, graph theory, probability,...

More particularly, in computer science, trees are a natural way to structure and manage the data, and as such, they are the basis of many crucial algorithms (binary search trees, quad-trees, 2-3-4 trees, ...). In this article, we are essentially interested in the random sampling of rooted planar trees. This topic itself is also subject to a extensive study. To mention only the best known algorithms, we can distinguish four approaches. The first two of them are in fact more general, but can be applied efficiently to the sampling of trees, the two others are ad hoc to tree sampling:

1. The random sampling by the recursive method [FZC94] of generating a tree from rules described with coefficients associated generating series [DPT10],
2. The random generation under Boltzmann model that allows uniform generation to approximate size from the evaluation of generating functions [DFLS04,BP10].
3. The random generation by Galton-Watson processes based on the dynamics of branching processes [Dev12],

4. The generation following Remy precepts.[ARS97a,ARS97b,BBJ13]

Concerning the generation of trees with a fixed degree sequence, the reference algorithms are due to Alonso et al. However, the complete understanding of their approach seems to us quite intricate. Moreover their approach is not optimal in terms of entropy (i.e. the minimum numbers of random bits necessary to draw an object uniformly as described in the famous Knuth-Yao paper [KY76]).

In this article, we give two versions of an algorithm for drawing efficiently trees whose the degree sequence is given. Our first version is fast and easy to implement, and its description is simple and (we hope) natural. It works, essentially like Alonso's algorithm, though we explicitly use the Lukasiewicz code of trees. Our second version only modifies the two first steps of the first algorithm. It is nearly optimal in terms of entropy because it uses only in average a linear number of random bits to draw a tree. Moreover, Lukasiewicz codes and a very elementary version of cyclic lemma allows us to give a simple proof of the theorem of Tutte [Tut64] which gives under an explicit multinomial form the number of plane trees with a given partition of the degrees.

From our sampler, we simulate various kind of trees. We focus our attention on unary-binary rooted planar trees (also called Motzkin trees) with a fixed frequency of unary nodes. In particular, we look for the variation of the height depending of the frequency of unary nodes. We can easily conjecture the nature of the variation.

Our second contribution is to describe and prove the distribution of the height according to the number of unary nodes. The proof follows a probabilistic approach and uses in a central the notion of continuous random trees (CRT). Even if the distribution of the height still follows a classical theta law, the expected value can leave the universal $\Theta(\sqrt{n})$ behaviours.

The general framework used in this paper to describe trees is the analytic combinatorics even if we use some classical notion on word theory and a basis of probabilistic concepts in the second part of the paper. More specifically, we deal with the symbolic method to describe the bijection between Lukasiewicz words and trees. A *combinatorial class* is a set of discrete objects $\mathcal{O}$, provided with a (multidimensional) *size function* $s :$ $\mathcal{O} \to \mathbb{N}^d$ for some integer d, in such way that for every $\boldsymbol{n} \in \mathbb{N}^d$, the set of discrete objects of size $\boldsymbol{n}$, denoted by $\mathcal{O}_{\boldsymbol{n}}$, is finite. In the classical definition, the size is just scalar, but for our parametrized problem this extension is more convenient. For more details, see for instance [FS09]. This approach is very well suited to the definition of trees. For instance, the class of binary trees $\mathcal{B}$ can be described by the following classical specification : $\mathcal{B} = \mathcal{Z} + \mathcal{Z}\mathcal{B}^2$.

In this framework, random sampling can be interpreted as follows. A *size uniform random generator* is an algorithm that generate discrete objects of a combinatorial class $(\mathcal{O}, s)$, such that for all objects $o_1, o_2 \in \mathcal{O}_{\boldsymbol{n}}$ of the same size, the probability to generate o_1 is equal to the probability to generate o_2.

The paper is organized as follows. Section 2 presents the definition of tree-alphabets, valid words, Lukasiewicz words ordered trees and the

links between the objects. Section 3 presents a re-description of an algorithm by Alonso et al. [ARS97a], using the notion of Lukasiewicz words. Our approach is to prove the algorithm step by step, using simple arguments. Section 4 present the dichotomic sampling method, which directly generates random valid words, using a linear number of random bits. The last part of the paper follows a simulate-guess-and-prove scheme. We first show some examples of random trees obtained from the generator. Then, we experimentally and theoretically study the evolution of the tree's height according to the proportion of unary nodes.

2   Words and Trees

2.1   Valid words and Lukasiewicz Words

This section is devoted to recall the one-to-one map between trees and Lukasiewicz words. This bijection is the central point for the sampling part of the paper. Let us recall basic definitions on words. An *alphabet* Σ is a finite tuple $(a_1, ..., a_d)$ of distinct symbols called *letters*. A word w defined on Σ is a sequence of letters from Σ. In the following, w_i denotes the i-th letters of the word w, $|w|$ its length and for all letter $a \in \Sigma$, $|w|_a$ counts the occurrences of the letter a in w. A *language* defined on Σ is a set of words defined of Σ.

The following new notion of *tree-alphabet* will make sense in the next sections. It will allow us to define subclasses of Lukasiewicz words will be in relation to natural combinatorial classes of trees.

Definition 1. *A* tree-alphabet *Σ_f is a couple (Σ, f) constituted by an alphabet $\Sigma = (a_1, \ldots, a_k)$ and a function $f : \Sigma \to \mathbb{N} \cup \{-1\}$ that associates each symbol of Σ to an integer such that:*
  i. $f(a_1) = -1$,
  ii. $f(a_i) \leq f(a_{i+1})$, for $1 \leq i < k$.

We finish this section by introducing Lukasiewicz words.

Definition 2. *A word w on the tree-alphabet $\Sigma_f = ((a_0, ..., a_k), f)$ is a f-Lukasiewicz word if :*
  i. for all $i < k$, we have $\sum_{j=0}^{i} |w|_{a_j} f(a_j) \geq 0$
  ii. $\sum_{i=1}^{k} |w|_{a_i} f(a_i) = -1$

When the condition ii. is verified, we say that the word w if f-valid. By extension and convenience, we also say that a k-tuple $(n_1, \ldots, n_k)$ is f-valid $\sum_{i=1}^{k} n_i f(a_i) = -1$.

The *Lukasiewicz words* $\mathcal{L}_f$ are just the union over all tree-alphabet Σ_f of the f-Lukasiewicz words.

A classical and useful representation of words on a tree-alphabet is to plot a path describing the evolution of $\sum_{j=1}^{i} f(w_j)$. Then, a word of size n is valid if and only if the path terminate at position $(n, -1)$ and it is a Lukasiewicz word if and only if the only step that goes under the $x - axis$ is the last one. In particular, these remarks prove that we can verify in linear time if a word is or not a Lukasiewicz word.

For instance, if $f(a) = -1$, $f(b) = 0$ and $f(c) = 1$, the following paths represent (from left to right) a Lukasiewicz word, a f-valid word and a non valid word:

$\Sigma_{j=1}^{i} f(w_i) + 1$

$c\ c\ b\ a\ b\ a\ a \qquad\qquad b\ a\ b\ a\ c\ c\ a \qquad\qquad c\ a\ c\ a\ b\ a\ a$
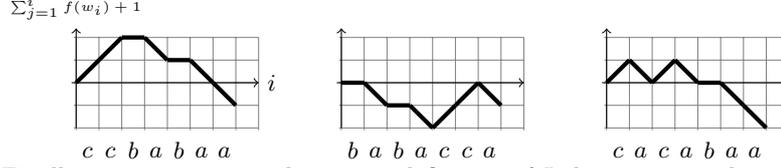
Finally, we can give an alternative definition of Lukasiewicz words in the framework of the symbolic method as follows: a word w defined over Σ_f is a Lukasiewicz word if $w = aw_1 \ldots w_{f(a)+1}n$ where $a \in \Sigma_f$ and $\forall i \leq f(a)+1$, w_i is a Lukasiewicz word. In other word, the combinatorial class of Lukasiewicz words follow the recursive specification:

$$L = \sum_{a \in \Sigma_f} aL^{f(s)+1}$$

2.2 The Tree classes

Rooted planar trees are very classical combinatorial objects. Let us recall how we can define them recursively and how this can be describe by a formal grammar. Let us begin by the rooted tree class $\mathcal{T}$ over the tree-alphabet Σ_f which can be define as the smallest set verifying:

  – $[x] \in \mathcal{T}$ for every $x \in \Sigma$ such that $f(x) = -1$.
  – Let x such that $f(x) = k$ and $T_1, \cdots, T_k$ in $\mathcal{T}$, then $x[T_1, \cdots, T_k]$ is in $\mathcal{T}$.

So, the set $\mathcal{T}$ of all planar Σ_f-*labelled trees* is a combinatorial class whose the size of a tree T is $(|f^{-1}(a_1)|, \cdots, |f^{-1}(a_d)|)$ where $\Sigma = (a_1, ..., a_d)$. And just observing the recursive definition, we can specify it from the following symbolic grammar:

$$G = \sum_{s \in \Sigma_f} sG^{f(s)+1}$$

Theorem 1 (Lukasiewicz). *The combinatorial class of f-Lukasiewicz words $\mathcal{L}_f$ is isomorphic to the combinatorial class of trees described by the specification (grammar) $G = \sum_{s \in \Sigma_f} sG^{f(s)+1}$.*

An explicit bijection can be done as follows: from a Σ_f-labelled tree T, a prefix walk gives a word. This word is a f-Lukasiewicz word. Conversely, from a f-Lukasiewicz word w, we build a tree recursively, the root is of degree $f(w_1) + 1$ and we continue with the sons as a left-first depth course.

3 A random sampler as a proof of Tutte's theorem

This section is devoted to describe the algorithm that we propose for drawing uniformly a rooted planar tree with a given sequence of degree. The diagram (Fig.1) shows the very simple strategy we adopt.
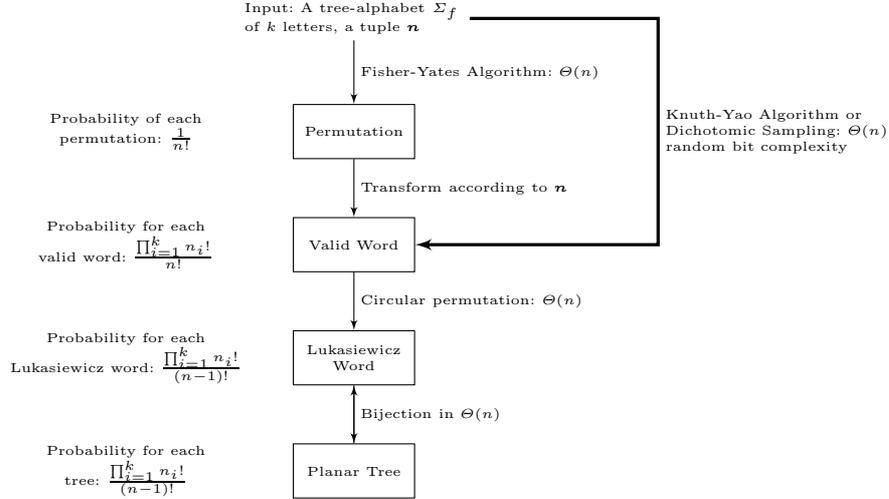
Fig. 1. Diagram of the two possible algorithms. The algorithm (Section 3) using the Fisher-Yates algorithm uses $\Theta(n \log n)$ random bits to generate a random tree with n nodes, but is easy to implement. The algorithm (Section 4) using the Knuth-Yao algorithm [KY76] or our dichotomic sampling method use a linear number of random-bit, but doesn't allow us to prove the Tutte's enumerative theorem.

The first algorithm contains 4 steps. The first and the last steps respectively consist in generating a random permutation using the Fisher-Yates algorithm and the transformation of a Lukasiewicz word into a tree. The two other steps are described in the two following subsections. Each subsection contains an algorithm, the proof of its validity, and its time and space complexity. We also uses the transformations to obtain enumeration results on each combinatorial object. Those enumeration results will be useful to prove that the random generator is size-uniform.

From a permutation to a valid word This part is essentially based on the following surjection from permutations to words. Consider the application Φ from Σ_n the set of permutations of size n to $\mathcal{W}_n$ the words of size n having for $1 \leq i \leq k, n_i$ letters a_i such that :

$$\Phi((\sigma_1, ..., \sigma_n)) = \phi(\sigma_1) \cdots \phi(\sigma_n)$$

where $\phi(k) = a_i$ if $n_1 + \cdots n_{k-1} + 1 \leq k \leq n_1 + \cdots n_k$.

Lemma 1. *For each valid word $w \in \Sigma_f^{\boldsymbol{n}}$ defined over a k letters alphabet, the number of permutation associated to w by the Algorithm 1 is exactly $\prod_{i=1}^{k} n_i!$.*

Proof. Let us define $m_i = \sum_{j=1}^{i-1} n_i$ and $m_1 = 0$. The application is invariant by permutation of the values inside $[m_i, \ldots, m_i + n_i]$. So, the cardinality of the kernel is $\prod_{i=1}^{k} n_i!$.

5

Algorithm 1: From a permutation to a valid word

Input: A tree-alphabet Σ_f of k letters and a tuple $\boldsymbol{n}$, a permutation σ of length n

Output: A tabular w encoding a valid word

1 Create a tabular w of size n;

2 $pos \leftarrow 0$;

3 **for** $i \in \{1, \dots, k\}$ **do**

4   **for** $j \in \{1, \dots, n_i\}$ **do**

5 $w[\sigma_{pos}] \leftarrow a_i$;

6 $pos \leftarrow pos + 1$;

7 **return** w;

Corollary 1. *The number of valid words in $\Sigma_f^{\boldsymbol{n}}$ is exactly $\frac{n!}{\prod_{i=1}^{k} n_i!}$.*

Lemma 2. *The time and space complexity of Algorithm 1 is $\Theta(n)$.*

Proof. The space complexity is linear since we create a tabular of size n. Instructions of line $1, 2, 5, 6$ can be done in constant time. Lines 5 and 6 are executed $\sum_{i=1}^{k} n_i$ times, that is to say n times.

From a valid word to a Lukasiewicz word This part is essentially based on a very simple version of the cyclic lemma which says that among the n circular permutations of a valid word, there is only one which is a Lukasiewicz word. Therefore, if we have a uniform random valid word and transform it into a Lukasiewicz word, we obtain a uniform Lukasiewicz word.
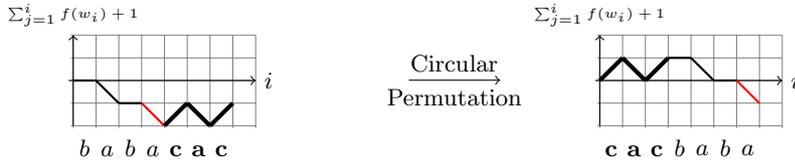
Fig. 2. An example: the valid word *babacac* is not a Lukasiewicz word but *cacbaba* is. The idea is to find the smallest value of i such that $\sum_{j=1}^{i} f(w_i)$ is minimal, and compute the word $w_{i+1} \cdots w_{|w|} w_1 \cdots w_i$

Lemma 3. *For each valid word $w \in \Sigma_f^{\boldsymbol{n}}$), there exists a unique integer ℓ such that $w_{\ell+1} \cdots w_n w_1 \cdots w_\ell$ is a Lukasiewicz word. Such integer is defined as the smallest integer that minimizes $\sum_{j=1}^{\ell} f(w_j)$.*

Proof. Let $w' = w_{\ell+1} \cdots w_n w_1 \cdots w_\ell$ be the *circular permutation* of w at a position ℓ. We notice that w' is a valid word. Let's now picture the

6

path representation of w and w' (see Figure 2). Let $b(i)$ (resp. $(a(i))$) be the height of the path at position i before (resp. after) the circular permutation. In other words:

$$b(i) = \sum_{j=1}^{i} f(w_j)$$

$$a(i) = \begin{cases} b(i) - b(\ell), & \text{for all } i \in \{\ell+1, \ldots, n\} \\ b(i) - b(\ell) - 1, & \text{for all } i \in \{1, \ldots, \ell\} \end{cases}$$

w' is a Lukasiewicz word iff $a(i) \geq 0$, for all $i \in \{1, \ldots, \ell-1, \ell+1, \ldots, n\}$, that is to say:

$$a(i) \geq 0 \iff \begin{cases} b(i) \geq b(\ell), & \text{for all } i \in \{\ell+1, \ldots, n\} \\ b(i) > b(\ell), & \text{for all } i \in \{1, \ldots, \ell-1\} \end{cases}$$

This concludes the proof.

Corollary 2. *The number of Lukasiewicz words in $\Sigma_f^{\boldsymbol{n}}$ is exactly $\frac{(n-1)!}{\prod_{i=1}^{k} n_i!}$.*

Proof. From Lemma 3 we know that each Lukasiewicz word can be obtained from exactly n valid words. We conclude using Corollary 1

Corollary 3 (Tutte). *The number of trees having n_i of type i and such that $(n_1, ..., n_k)$ is f-valid is exactly $\frac{(n-1)!}{\prod_{i=1}^{k} n_i!}$.*

Proof. It is a direct consequence of the bijection between trees and Lukasiewicz words.

We use the property of Lemma 3 to describe an algorithm that transforms any valid word into its associated Lukasiewicz word.

Lemma 4. *Algorithm 2 transforms a valid word into its Lukasiewicz word. Its time and space complexity is $\Theta(n)$.*

Proof. The space complexity is linear since we create a tabular v of size n. The first loop computes unique integer ℓ such that $w_{\ell+1} \cdots w_n w_1 \cdots w_\ell$ is a Lukasiewicz word, in linear time. The second and the third loop fill tabular the v of length n such that $v = w_{\ell+1} \cdots w_n w_1 \cdots w_\ell$.

3.1 First algorithm

Theorem 2. *Algorithm 3 is a random planar tree generator. Its time and space arithmetic complexity is linear.*

Algorithm 2: From a valid word to a Lukasiewicz word

Input: A valid word w of length n according to (Σ, f, occ)
Output: A tabular v encoding a Lukasiewicz word

1 $min \leftarrow cur \leftarrow f(w_1)$;
2 $\ell \leftarrow 1$;
3 for $i \in \{2, \ldots, n\}$ **do**
4 $cur \leftarrow cur + f(w_i)$;
5   **if** $cur < min$ **then**
6 $\ell \leftarrow i$;
7 $min \leftarrow cur$;

8 Create a tabular v of length n;
9 for $i \in \{1, \ldots, \ell\}$ **do**
10 $v[i + \ell + 1] \leftarrow w[i]$;
11 for $i \in \{\ell + 1, \ldots, n\}$ **do**
12 $v[i - \ell - 1] \leftarrow w[i]$;
13 return v;

Algorithm 3: Random Planar Tree Generator

Input: A tree-alphabet Σ_f of k letters and a tuple $\boldsymbol{n}$
Output: A random planar tree satisfying Σ_f and $\boldsymbol{n}$

1 Generate a random permutation σ using Fisher-Yates Algorithm;
2 Transform σ into a valid word w;
3 Transform w into a Lukasiewicz word v;
4 Transform v in a planar tree t
5 return t;
---

4 The dichotomic sampling method

Using the diagram of Figure 1 above, we arrive at the algorithm 3. However, this algorithm is not optimal in random bit because drawing the permutation consumes more bits than necessary. We shall describe another method to more efficiently generate a valid word. The problem is just to draw a f-valid word from a f-valid tuple $\boldsymbol{n} = (n_1, \ldots, n_k)$. For that purpose, consider the random variable A on the letters of Σ, assume that A_1 follows the distribution $D_{\boldsymbol{n}}$: $Prob(A_1 = a_i) = \dfrac{n_i}{\sum_i n_i}$, draw A_1 (says $A_1 = a_j$) and put it in the first place in the word (i.e. $w_1 = a_j$). Now, A_2 is conditioned by A_1, just by decrease by one n_j, again draw A_2 and put it in the second place, and so on. This algorithm is described below (see Algorithm 4). It is clear that the built word is a f-valid word, because it contains exactly the good number of each letters. Now, it is drawn uniformly, indeed, in a uniform f-valid word, the first letter follows exactly the distribution $D_{\boldsymbol{n}}$, the sequel follows directly by induction.

Algorithm 4: From a tuple $\boldsymbol{n}$ to a valid word

Input: A tree-alphabet Σ_f of k letters and a tuple $\boldsymbol{n}$
Output: A tabular w encoding a valid word

1 Create a tabular w of size n;
2 **for** $i \in \{1, \ldots, n\}$ **do**
3 $k \leftarrow Distrib(\boldsymbol{n})$ (k is drawn according to the distribution $D_{\boldsymbol{n}}$);
4 $w[i] \leftarrow a_k$;
5 $\boldsymbol{n} \leftarrow \boldsymbol{n} - \boldsymbol{e}_k$ ($\boldsymbol{e}_k$ denotes the k-th canonical vector);
6 **return** w;

So, to obtain a random-bit optimal sampler, we just need to have a optimal sampler for general discrete distribution. But, it is exactly the result obtained by Knuth-Yao [KY76]. This achieves the proof of:

Theorem 3. *The algorithm described below is a random-bit optimal sampler for rooted planar tree with a given sequence of degree.*

Nevertheless, according to the authors, the Knuth-Yao algorithm can be inefficient in practice (because it needs to solve the difficult question to generate infinite DDG-trees). There is a long literature on it which is summarized in the book of L. Devroye [?]. Let just mention the interval sampler from [HH97] and the alias methods [Vos91,Wal77,MTW04].

We propose in the sequel a nearly optimal and very elementary algorithm, called *dichotomic sampling*, to draw a random variable X following a given discrete distribution of k parts, say, $Prob(X = x_i) = p_i$ for $1 \leq i \leq k$. Let us denote $s_j = \sum_{i \leq j} p_i$ and let us express X as $[s_1, \cdots, s_{k-1}]$. Now, let m be the largest integer such that $s_m \leq 1/2$ and $m = 0$ if there is no such a number. Then to draw $X_1 = [s_1, \ldots, s_{k-1}]$, we can flip a coin (Bernoulli choice of parameter $1/2$), and if the result is heads then if

$m = 0$ return x_0 or draw in $X_2 = [2s_1, \ldots, 2s_m]$, if the result is tails and if $m = k - 1$ then return x_k else draw in $X_2 = [2s_{m+1} - 1, \ldots, 2s_{k-1} - 1]$ and continue recursively while $X_\mu \neq \emptyset$

The dichotomic sampling algorithm implies the following induction for C_n the mean number of flip needed for drawing when there are $n + 1$ parts : $C_1 = 2$ and $C_k = 1 + \frac{1}{2}\max_{0 \leq k \leq m}(C_m + C_{k-m})$. First, let us assume that C_k is concave, so let us consider $\tilde{C}_k = 1 + \frac{1}{2}(\tilde{C}_{\lfloor \frac{k}{2} \rfloor} + \tilde{C}_{\lceil \frac{k}{2} \rceil})$. A short calculation shows that $\tilde{C}_n = \lfloor \ln_2(n-1) \rfloor + 1 + \dfrac{n}{2^{\lfloor \ln_2(n-1) \rfloor}}$. Now, by induction, we can easy check that $C_k = \tilde{C}_k$. So, in particular, $C_k \leq 2 + \ln_2(k)$.
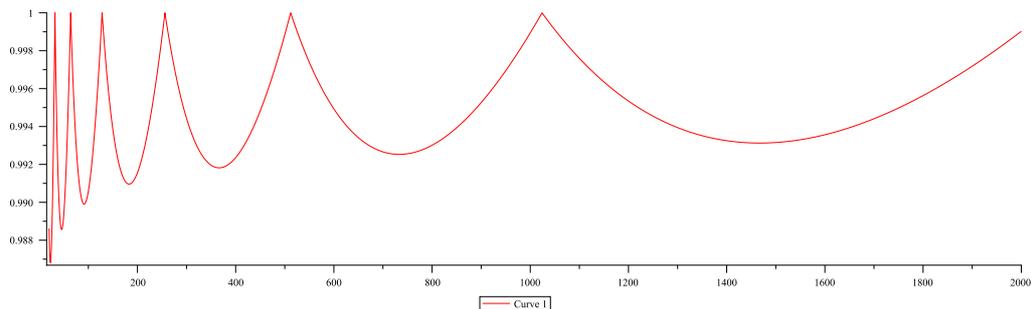
Fig. 3. Graphic for Mean Cost $\dfrac{C_k}{2 + \ln_2(k)}$

Note that the sequence C_k can also be analyzed by classical Mellin transform techniques and the periodic phenomena we show in the figure 3 is quite familiar.

5 Simulate-Guess-and-prove : Analysis of height

In this section, we study experimentally and theoretically the height of random Motzkin trees (unary-binary) when the proportion of unary nodes fluctuates.

Figure 4 shows example of random Motzkin trees generated with the algorithm from Section 3, with different proportions of unary nodes. Figure 5 shows the evolution of the height of trees when one increases the proportions of unary nodes.

In the following, we study the height of Motzkin trees according to the proportion of unary nodes, using exclusively probabilistic arguments.

The continuum random tree (CRT) is a random continuous tree defined by Aldous [Ald93], which is closely related to Brownian motion. In particular, the height of the CRT has the same law as the maximum of a Brownian excursion. The CRT can be viewed as the renormalized limit of

10
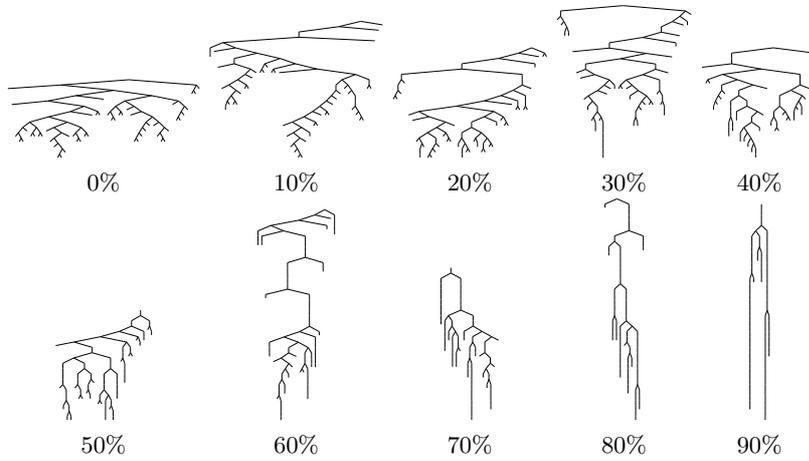
Fig. 4. Example of Motzkin trees with 101 nodes generated with our algorithm, where the proportion of unary nodes varies from 0% to 90%.
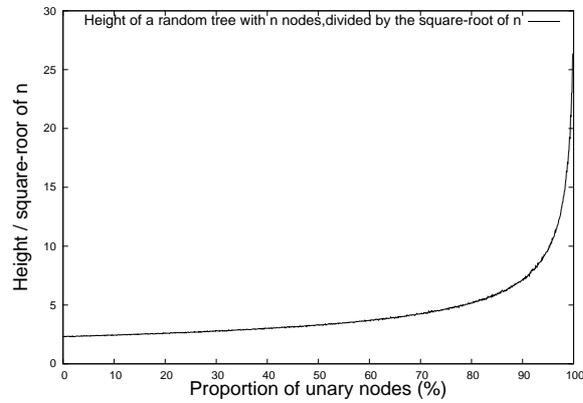
Fig. 5. In this example, all random trees have $n = 1000$ nodes. For each proportion of unary nodes, varying from 0 to 99, 9 percent, 10 000 Motzkin trees have been generated. The curve shows how the average height of Motzkin trees, divided by the square-root of n.

11

several models of large trees, in particular, critical Galton-Watson trees with finite variance conditioned to have a large population [GK98,Duq]. Our model does not exactly fit into this framework, however, it is quite clear that the proofs can be adapted to our situation. We show here a convergence result related to the height of Motzkin trees.

Theorem 1 *Let $(c_n, n \geq 1)$ be a sequence of integers such that $c_n = o(n)$ and $(\log n)^2 = o(c_n)$. Then one can construct, on a single probability space, a family $(T_n, n \geq 1)$ of random trees and a random variable $H > 0$ such that*
(i) for every $n \geq 1$, T_n is a uniform Motzkin tree with n vertices and $c_n + 1$ leaves.
(ii) H has the law of the height of the CRT
(iii) almost surely,
$$\frac{\sqrt{c_n}}{n} height(T_n) \to H$$

Proof. The proof's idea is the following:
  − A Motzkin tree can been seen as a binary tree with $2c_n + 1$ nodes in which we each node can be replaced by a sequence of unary nodes. If n is the size of the Motzkin tree, then the number of unary node is $n - 2c_n - 1$.
  − The height of a leaf in the Motzkin tree is equal its length in the binary tree plus the lengths of the sequences of unary nodes between the leaf and the tree's root.
  − We study the probability that the lengths sum of the sequences of unary nodes between a given leaf and the tree's root is equal to a given value.
  − We use this result to frame the generic height of T_n.

We assume that $(c_n, n \geq 1)$ is non-decreasing, otherwise, the proof can be easily adapted. Let us call the skeleton of a Motzkin tree the binary tree obtained by forgetting the vertices having one child. Denote by S_n the skeleton of T_n. For a leaf l, let $d(l)$ be the distance of l to the root in S_n and $D(l)$ the distance of l to the root in T_n.

First, one can construct the sequence $(S_n, n \geq 1)$ by Rémy's algorithm [?] and it can be shown that S_n converges in a strong sense to a CRT [CHar], in particular,
$$\frac{height(S_n)}{\sqrt{c_n}} \to H$$
where H has the law of the height of the CRT.

Next, for every $n \geq 1$, one can obtain T_n from S_n by replacing each edge e of S_n with a "pipe" containing X_e nodes of degree 2. The family (X_e) is a $2c_n$-dimensional random vector with non-negative integer entries, and it is uniformly distributed over all vectors of this kind such that the sum of the entries is $n - 2c_n - 1$. Let us denote $(X_e) = (X_1, \ldots, X_{2c_n})$ (we should write $(X_1^{(n)}, \ldots, X_{2c_n}^{(n)})$ but we want to make the notation lighter). It is a classical remark that the random variable $(X_1, \ldots, X_{2c_n})$ has the same law as $(Y_1, \ldots, Y_{2c_n})$ conditional on the event $\sum_i Y_i = n - 2c_n - 1$, where the Y_i are independent, geometric random variables with mean
$$m_n = \frac{n - 2c_n - 1}{2c_n}$$

Moreover, since the sum $\sum_i Y_i$ has mean $n - 2c_n - 1$ and variance $\sim c_n m_n^2$, a classical local limit theorem [Gne48] tells us that there exists a constant $c > 0$ such that for every $n \geq 1$,

$$\mathbb{P}(\sum_i Y_i = n - 2c_n - 1) \geq \frac{1}{c\sqrt{c_n}m_n} \tag{1}$$

Fix $\varepsilon > 0$. Pick at random a realization of Rémy's algorithm, yielding a sequence of binary trees $(S_n, n \geq 1)$ such for every $n \geq 1$, S_n, has $c_n + 1$ leaves. Then almost surely, there exists $H > 0$ such that the height of S_n, which we denote h_n, satisfies

$$\frac{h_n}{\sqrt{c_n}} \to H \tag{2}$$

From now on, since we have chosen our sequence $(S_n, n \geq 1)$, the symbols $\mathbb{P}$ and $\mathbb{E}$ will refer to the probability and expectation with respect to the random variables (X_i), (Y_i), (Z_i).

If a leaf l in S_n is at a distance $d(l)$ from the root, then its distance $D(l)$ from the root in T_n is the sum of $d(l)$ random variables in the family (X_e). Therefore,

$$\begin{aligned}
\mathbb{P}(D(l) = k) &= \mathbb{P}(X_1 + \ldots + X_{d(l)} = k) \\
&= \mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k | \sum_i Y_i = n - 2c_n - 1) \\
&= \frac{\mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k, \sum_i Y_i = n - 2c_n - 1)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)} \\
&\leq \frac{\mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)}
\end{aligned}$$

The right-hand side is maximized when $d(l) = h_n$. We shall now use independent, exponential random variables $(Z_1, \ldots, Z_{2c_n})$ with mean

$$\mu_n = \frac{1}{\log(m_n/(m_n - 1))} \tag{3}$$

It is easy to check that for every integer $k \geq 0$,

$$\mathbb{P}(Z_1 \in [k, k+1]) = \mathbb{P}(Y_1 = k).$$

Therefore, we can define Y_i as the integer part of Z_i for each i. Since $Z_i \geq Y_i$ for each i,

$$\mathbb{P}\left(\frac{Y_1 + \ldots + Y_{h_n}}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right) \leq \mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n}}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right)$$

Substracting the expectation,

$$\begin{aligned}
&\mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n}}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right) \\
&= \mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n\mu_n}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H - \frac{h_n\mu_n}{\sqrt{c_n}m_n}\right)
\end{aligned}$$

13

Because of (3) and (2), we have, for n large enough,

$$\left(1 - \frac{\varepsilon}{2}\right) H \leq \frac{h_n \mu_n}{\sqrt{c_n} m_n} \leq \left(1 + \frac{\varepsilon}{2}\right) H$$

This entails that for n large enough,

$$(1 + \varepsilon) H - \frac{h_n \mu_n}{\sqrt{c_n} m_n} \leq \frac{\varepsilon H}{2}$$

and therefore,

$$\mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n \mu_n}{\sqrt{c_n} m_n} \geq (1 + \varepsilon) H - h_n \mu_n\right)$$

$$\leq \mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n \mu_n}{\sqrt{c_n} m_n} \geq \frac{\varepsilon H}{2}\right)$$

We now use the Laplace transform: for every $\lambda > 0$,

$$\mathbb{E} \exp(\lambda Z_1 - \mu_n) = \frac{e^{-\lambda \mu_n}}{1 - \lambda \mu_n}$$

The Markov inequality yields

$$\mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n \mu_n}{\sqrt{c_n} m_n} \geq \frac{\varepsilon H}{2}\right) \leq \left(\frac{e^{-\lambda \mu_n}}{1 - \lambda \mu_n}\right)^{h_n} \exp\left(-\lambda \sqrt{c_n} m_n \frac{\varepsilon H}{2}\right)$$

Let (t_n) be a sequence of positive real numbers such that t_n tends to 0 and that $\sqrt{c_n} t_n / \log n$ tends to infinity. Choose λ such that $\lambda \mu_n = t_n$. Then,

$$\mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n \mu_n}{\sqrt{c_n} m_n} \geq \frac{\varepsilon H}{2}\right) \leq \left(\frac{e^{-t_n}}{1 - t_n}\right)^{h_n} \exp\left(-\frac{t_n \sqrt{c_n} m_n \varepsilon H}{2 \mu_n}\right)$$

For n large enough, we have $m_n \geq \mu_n / 2$ and

$$\frac{e^{-t_n}}{1 - t_n} \leq 1 + 2 t_n^2$$

Therefore, for n large enough

$$\mathbb{P}\left(\frac{Z_1 + \ldots + Z_{h_n} - h_n \mu_n}{\sqrt{c_n} m_n} \geq \frac{\varepsilon H}{2}\right) \leq (1 + 2 t_n^2)^{h_n} \exp\left(-\frac{\varepsilon H t_n \sqrt{c_n}}{4}\right)$$

Summing up, if n is large enough, then for every leaf l,

$$\mathbb{P}\left(\frac{D(l)}{\sqrt{c_n} m_n} \geq (1 + \varepsilon) H\right) \leq \frac{(1 + 2 t_n^2)^{h_n} \exp\left(-\frac{\varepsilon H t_n \sqrt{c_n}}{4}\right)}{\mathbb{P}(\sum_i Y_i = n - 2 c_n - 1)}$$

Using the estimate (1),

$$\mathbb{P}\left(\frac{D(l)}{\sqrt{c_n} m_n} \geq (1 + \varepsilon) H\right) \leq c \sqrt{c_n} m_n (1 + 2 t_n^2)^{h_n} \exp\left(-\frac{\varepsilon H t_n \sqrt{c_n}}{4}\right)$$

14

Since there are $c_n + 1$ leaves, and since the probability of the union is less that the sum of the probabilities, for n large enough,

$$\mathbb{P}\left(\frac{height(T_n)}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right) \leq c(c_n+1)\sqrt{c_n}m_n(1+2t_n^2)^{h_n}\exp\left(-\frac{\varepsilon H t_n\sqrt{c_n}}{4}\right)$$

The upper bound can be rewritten as

$$c\exp\left(h_n\log(1+2t_n^2) - \frac{\varepsilon H t_n\sqrt{c_n}}{4} + \log m_n + \frac{3}{2}\log(c_n+1)\right)$$

Recall that for n large enough,

$$h_n \leq (1+\varepsilon/2)H\sqrt{c_n}$$

and then our bound becomes

$$\exp\left(H\sqrt{c_n}\left[(1+\varepsilon/2)\log(1+2t_n^2) - \frac{\varepsilon t_n}{4}\right] + \log m_n + \frac{3}{2}\log(c_n+1)\right)$$

Since $t_n \to 0$, for n large enough,

$$[(1+\varepsilon/2)\log(1+2t_n^2) - \frac{\varepsilon t_n}{4}] \geq -\frac{\varepsilon t_n}{8}$$

and so for n large enough, our bound becomes

$$b_n = \exp\left(\frac{-H\varepsilon t_n\sqrt{c_n}}{8} + \log m_n + \frac{3}{2}\log(c_n+1)\right)$$

Now because of the assumption that $\sqrt{c_n}t_n/\log n \to \infty$, we remark that $\sum b_n < \infty$. Thus by the Borel-Cantelli lemma, almost surely, conditional on the sequence (S_n), for n large enough,

$$\frac{height(T_n)}{\sqrt{c_n}m_n} \leq (1+\varepsilon)H$$

Integrating with respect to the law of the sequence (S_n), we find that almost surely, there exists a random variable H which has the law of the height of the CRT and such that for n large enough,

$$\frac{height(T_n)}{\sqrt{c_n}m_n} \leq (1+\varepsilon)H$$

Likewise, one shows that almost surely, for n large enough,

$$\frac{height(T_n)}{\sqrt{c_n}m_n} \geq (1-\varepsilon)H$$

This being true for every positive ε, our result is established.

Remark In the case when the number of leaves is proportional to the number of vertices, $c_n \sim kn$ for some constant $k \in (0, 1/2]$, it can be shown by the same arguments that $\frac{height(T_n)}{\sqrt{n}}$ converges to $2(1-k)H$.
In the case when $(\log n)^2/c_n$ does not tend to 0, a refinement in the proof is necessary. Typically, replacing the inequality (1) with a stochastic domination argument would prove that the height of the tree converges in

15

distribution whenever $c_n \to \infty$. To prove an almost sure convergence, a more detailed construction would be needed.

General case

We only assume that c_n tends to infinity. The construction of the skeleton and the convergence of Rémy's algorithm still hold. The representation of the variables X_i as conditioned versions of the Y_i can be refined in the following manner:

$$\mathbb{P}(X_1 + \ldots + X_{d(l)} \geq A)$$

$$= \mathbb{P}(Y_1 + \ldots + Y_{d(l)} \geq A | \sum_i Y_i = n - 2c_n - 1)$$

$$= \sum_{k=A}^{\infty} \mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k | \sum_i Y_i = n - 2c_n - 1)$$

$$= \sum_{k=A}^{\infty} \mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k | \sum_{i=d(l)}^{2c_n} Y_i = n - 2c_n - 1 - k)$$

$$= \sum_{k=A}^{\infty} \frac{\mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k, \sum_{i=d(l)}^{2c_n} Y_i = n - 2c_n - 1 - k)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)}$$

$$= \sum_{k=A}^{\infty} \frac{\mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k)\mathbb{P}(\sum_{i=d(l)}^{2c_n} Y_i = n - 2c_n - 1 - k)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)}$$

Gnedenko's result also gives the existence of a real C such that for every integer k,

$$\mathbb{P}(\sum_{i=d(l)}^{2c_n} Y_i = n - 2c_n - 1 - k)) \leq \frac{C}{\sqrt{c_n - d(l)}m_n} \tag{4}$$

From (1) and (4) we deduce that if $d(l) \leq c_n/2$, the following stochastic domination bound hols:

$$\mathbb{P}(Y_1 + \ldots + Y_{d(l)} \geq A | \sum_i Y_i = n - 2c_n - 1)$$

$$= \sum_{k=A}^{\infty} \frac{\mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k)\mathbb{P}(\sum_{i=d(l)}^{2c_n} Y_i = n - 2c_n - 1 - k)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)}$$

$$\leq \frac{C\sqrt{2}}{c} \sum_{k=A}^{\infty} \mathbb{P}(Y_1 + \ldots + Y_{d(l)} = k)$$

To sum up, if $d(l) \leq c_n/2$,

$$\mathbb{P}(X_1 + \ldots + X_{d(l)} \geq A) \leq \frac{C\sqrt{2}}{c}\mathbb{P}(Y_1 + \ldots + Y_{d(l)} \geq A) \tag{5}$$

Recall that for every leaf l of S_n, $d(l) \leq h_n$, and that because of (2), the condition $d(l) \leq c_n/2$ is satisfied for all leaves if n is large enough. The bound using conditioning gave

$$\mathbb{P}\left(\frac{D(l)}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right) \leq \frac{(1 + 2t_n^2)^{h_n} \exp\left(-\frac{\varepsilon H t_n \sqrt{c_n}}{4}\right)}{\mathbb{P}(\sum_i Y_i = n - 2c_n - 1)}$$

16

But using the stochastic domination bound (5), we can improve this to

$$\mathbb{P}\left(\frac{D(l)}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right) \leq \frac{C\sqrt{2}}{c}(1+2t_n^2)^{h_n}\exp\left(-\frac{\varepsilon H t_n \sqrt{c_n}}{4}\right)$$

for n large enough. Taking $t_n = c_n^{-1/4}$ and using (2), we find that the probability

$$\mathbb{P}\left(\frac{D(l)}{\sqrt{c_n}m_n} \geq (1+\varepsilon)H\right)$$

tends to 0 as n goes to infinity, for every positive ε. Likewise, if e_n is a leaf in S_n such that $d(l) = h_n$, one can prove that the probability

$$\mathbb{P}\left(\frac{D(e_n)}{\sqrt{c_n}m_n} \leq (1-\varepsilon)H\right)$$

goes to 0 as n goes to infinity. This proves that

$$\frac{height(S_n)}{\sqrt{c_n}}$$

converges in distribution to H. So we have the more general result

Theorem 2 *Let $(c_n, n \geq 1)$ be a sequence of integers such that $c_n \to \infty$ as $n \to \infty$. Let $(T_n, n \geq 1)$ be a family of random trees such that for every $n \geq 1$, T_n is a uniform Motzkin tree with n vertices and $c_n + 1$ leaves. Then*

$$\frac{\sqrt{c_n}}{n}height(T_n)$$

converges in distribution to the law of the height of a CRT.

6   Conclusion

In this paper, we gave a new sampler for of rooted planar trees that satisfies a given partition of degrees. This sampler is now optimal in terms of random bit complexity. We apply it to predict the average height of a random Motzkin in function of its frequency of unary nodes. We then prove some unconventional height phenomena (i.e. outside the universal $\Theta(\sqrt{n})$ behaviour. Our work can certainly be extended to more complicate properties than the list of degrees. We can expected similar samplers for simple patterns constraints, such as fixed numbers of paths of fixed length, or fixed numbers of complete trees.

References

[Ald93] David Aldous. The continuum random tree. iii. *Ann. Probab.*, 21(1):248–289, 1993.

[ARS97a] Laurent Alonso, Jean-Luc Remy, and René Schott. A linear-time algorithm for the generation of trees. *Algorithmica*, 17(2):162–183, 1997.

[ARS97b] Laurent Alonso, Jean-Luc Remy, and René Schott. Uniform generation of a schröder tree. *Inf. Process. Lett.*, 64(6):305–308, 1997.

[BBJ13] Axel Bacher, Olivier Bodini, and Alice Jacquot. Exact-size sampling for motzkin trees in linear time via boltzmann samplers and holonomic specification. In Markus E. Nebel and Wojciech Szpankowski, editors, *ANALCO*, pages 52–61. SIAM, 2013.

[BP10] Olivier Bodini and Yann Ponty. Multi-dimensional Boltzmann Sampling of Languages. In *DMTCS Proceedings*, number 01 in AM, pages 49–64, Vienne, Autriche, 2010. 12pp.

[CHar] N. Curien and B. Haas. The stable trees are nested. *Prob. Theory Rel. Fields*, to appear.

[Dev12] Luc Devroye. Simulating size-constrained galton-watson trees. *SIAM J. Comput.*, 41(1):1–11, 2012.

[DFLS04] Philippe Duchon, Philippe Flajolet, Guy Louchard, and Gilles Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability & Computing*, 13(4-5):577–625, 2004.

[DPT10] Alain Denise, Yann Ponty, and Michel Termier. Controlled non uniform random generation of decomposable structures. *Theoretical Computer Science*, 411(40-42):3527–3552, 2010.

[Duq] T. Duquesne. A limit theorem for the contour process of conditioned galton-watson trees.

[FS09] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[FZC94] Philippe Flajolet, Paul Zimmermann, and Bernard Van Cutsem. A calculus for the random generation of labelled combinatorial structures. *Theor. Comput. Sci.*, 132(2):1–35, 1994.

[GK98] J. Geiger and G. Kersting. The galton-watson tree conditioned on its height. *Proceedings 7th Vilnius Conference.*, 1998.

[Gne48] B. V. Gnedenko. On a local limit theorem of the theory of probability. *Uspehi Matem. Nauk (N. S.)*, 3(3(25)):187–194, 1948.

[HH97] Te Sun Hao and M. Hoshi. Interval algorithm for random number generation. *Information Theory, IEEE Transactions on*, 43(2):599–611, 1997.

[KY76] Donald E. Knuth and Andrew C. Yao. The Complexity of Nonuniform Random Number Generation. In J. F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results*. Academic Press, New York, 1976.

[MTW04] George Marsaglia, Wai Wan Tsang, and Jingbo Wang. Fast generation of discrete random variables. *Journal of Statistical Software*, 11(3):1–11, 7 2004.

[Tut64]   W. T. Tutte. The number of planted plane trees with a given partition. *The American Mathematical Monthly*, 71(3):pp. 272–277, 1964.

[Vos91]   Michael D. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering*, 17(9):972–975, 1991.

[Wal77]   Alastair J. Walker. An Efficient Method for Generating Discrete Random Variables with General Distributions. *ACM Transactions on Mathematical Software*, 3(3):253–256, September 1977.