Extended boxed product and application to synchronized trees

Olivier Bodini[*], Antoine Genitrini[†] and Nicolas Rolin[*].
{Olivier.Bodini;Nicolas.Rolin}@lipn.univ-paris13.fr and Antoine.Genitrini@lip6.fr.

January 7, 2017

We introduce a new technique to specify increasing labeled structures. For such objects, the boxed product as introduced by Greene is sufficient to efficiently specify the class, however for particular classes the size of the specification is very large. In particular, in the case of partially ordered sets, the calculus of the total orders compatibles with the poset (called linear extensions) can be tedious. We here developed an idea due to Stanley that uses a geometrical interpretation to calculate the linear extensions of a given poset. We will present a way to extent this idea to the symbolic method, and illustrate it with the example of specific increasing trees with exactly one label repeated, and show how to uniformly generate such structures.

1 Introduction

In 1983, Greene [Gre83] developed new approaches to address labeled combinatorial structures that require some order constrained for the labeling. In particular, he introduced the binary boxed operator $\mathcal{A}^{\square} \star \mathcal{B}$ that encodes the labeled product of the classes $\mathcal{A}$ and $\mathcal{B}$ satisfying the fact that the smallest label appears in the component of $\mathcal{A}$. From the standard convolution issued in the product of the related exponential generating functions $A(z)$ and $B(z)$, Greene deduced that the generating function for $\mathcal{A}^{\square} \star \mathcal{B}$ is given by

$$\int_0^z A'(t) \cdot B(t) dt.$$

This now classical constrained product has been fully integrated into the *symbolic method* (cf. [FS09]) and many structures are based on this simple order constraint. For instance, specific subclasses of permutations (cf. several examples and references in [FS09, II.6.3.]), increasingly labeled trees [BFS92] or partially increasingly labeled trees [BGP13] can be specified thanks to the boxed product.

In the random sampling domain, the boxed product also appears as a cornerstone for the efficiency of the recursive method [FZVC94] [NW78]. Although other elegant specifications based on such ideas are presented by Flajolet and Sedgewick, it seems clear that some more intricate problems cannot be encoded with only such a global order constraint. The class of increasingly labeled directed acyclic graphs contains much more complex dependencies than we can expect to simply encode with only boxed products.

More specifically, let us consider two directed sequences of atoms, add a top node that is the common ancestor of both sequences and a bottom node that is the common descendant of both

[*]Laboratoire d'Informatique de Paris-Nord, CNRS UMR 7030 - Institut Galilée - Université Paris-Nord, 99, avenue Jean-Baptiste Clément, 93430 Villetaneuse, France.

[†]Laboratoire d'Informatique de Paris 6, CNRS UMR 7606 and Université Pierre et Marie Curie, 4 place Jussieu, 75005 Paris, France.

sequences, the boxed product seems to be inappropriate in order to simply describe an increasing labeling of such a structure. Even (as we can see in the sequel) if this structures can in fact be specified by using boxed products, such a specification is of exponential size in the number of atoms of both sequences. The main goal of this paper is to introduce new more tractable operators.

In the context of posets arises the question to count the number of total orders compatible with a given poset, Stanley [Sta86b] introduced for this a nice geometrical interpretation. The vertices of the poset define a unit hypercube in which each total order corresponds to a specific polytope in this hypercube. By consequence, the number of compatible orders is directly related to a volume that can be computed by multivariate integration. We can for example encode the previous example (the shuffling of the two sequences with a source and a sink) with Stanley's method and thus translate it into an equation satisfied by the generating function. With this approach, we avoid the size explosion of the specification. In the same vein, for several cases, this approach allows us to determine an exact formula for the generating function.

So, in this paper, we present a way to integrate Stanley's approach in the context of symbolic method and we illustrate the whole process on a specific and paradigmatic example: The enumeration of the number of increasing trees with one repetition among the labels, which we will call *synchronized trees*.

In a second step, we also adapt the method to be applied to recursive sampling [FZVC94] of the structures under consideration. Finally, we conclude the paper by mentioning a much more intricate perspective based on some integral properties: even if the structures we are considering have very specific substructures (e.g. a mix between some increasingly labeling and some labeling without order constraint) the approach is still valid and gives a possible way for the enumeration and the uniform random sampling problem of such specific structures.

2 Enriched Stanley's approach

From 1986 in his paper [Sta86b] and then in his book [Sta86a, chapter 3], Stanley developed a strategy in order to enumerate the number of linear extensions, or chains, satisfying a partially ordered set. To achieve this goal, he managed to give a geometrical interpretation of a poset and its related linear extensions set as a subspace of a unit hypercube and some polytopes that partitioned it. Thus the number of linear extensions of a poset is reduced to the calculation of an integration in the hypercube (in several variables). Recall that the exact computation of the number of linear extensions of a poset has been proved #P complete by Brightwell and Winkler [BW91]. The difficulty of counting has been transferred by Stanley in the exact computation of a multiple integral.

Let us now present the adaptations we proceed to Stanley's approach in order to extract much more information than only the number of linear extensions. In fact, by inserting some formal variables in the integrations we manage to partition the linear extensions according to some of their structural properties. As a trivial example, we can enumerate binary increasing trees according to the sizes of two subtrees attached to the root.

For the rest of the paper, we use the trivial total order $\leq$ for the integers $\{1, .., n\}$ whatever a given positive integer n.

Definition 2.1. *A partial order $\leq_\alpha$ (or poset) on E is a relation between some elements of E such that:*

- $\leq_\alpha$ *is* reflexive*:* $\forall x \in E, x \leq_\alpha x$
- $\leq_\alpha$ *is* antisymmetric*:* $\forall x, y \in E, x \leq_\alpha y, y \leq_\alpha x \Rightarrow x = y$
- $\leq_\alpha$ *is* transitive*:* $\forall x, y, z \in E, x \leq_\alpha y, y \leq_\alpha z \Rightarrow x \leq_\alpha z$

We define the *size* of a poset as its number of elements. We recall that a *linear extension* (or chain) of a partial order $\leq_A$ is a total order compatible with $\leq_\alpha$, i.e an order $\leq_A$ on E such that $\forall x, y \in E, x \leq_\alpha y \Rightarrow x \leq_A y$. We note $\mathcal{LE}(\leq_\alpha)$ the set of linear extensions of $\leq_\alpha$.

The first adaptation of Stanley's strategy consists in seeing the unit hypercube as a formal hypercube $[0,z]^n$, with z a variable in $[0,1]$, and using it to be the integration domain. It will allow us to extend Stanley's study to generating functions. Obviously, by evaluating $z=1$, we get back all Stanley's result.

Definition 2.2. *Let $\leq_\alpha$ be a partial order over a finite set of size n, and z a variable in $[0,1]$. We define $E_{\leq_\alpha}(z) = \{(z_1,\dots,z_n) \in [0,z]^n \mid \forall i \neq j \in \{1,\dots,n\},\ i \leq_\alpha j \Rightarrow z_i \leq z_j\}$ as the domain associated to $\leq_\alpha$.*

The main result of Stanley for the enumeration of the linear extensions is the following:

$$|\mathcal{LE}(\leq_\alpha)| = n! \cdot \mathrm{Vol}(\leq_\alpha),$$

where $\mathrm{Vol}(\leq_\alpha)$ is the volume of the polytope associated to $\leq_\alpha$.

Definition 2.3. *Let $\leq_\alpha$ be a partial order over a finite set of size n, and z a variable in $[0,1]$. We define $E_{\leq_\alpha}(z) = \{(z_1,\dots,z_n) \in [0,z]^n \mid \forall i \neq j \in \{1,\dots,n\},\ i \leq_\alpha j \Rightarrow z_i \leq z_j\}$ as the domain associated with $\leq_\alpha$.*

By using this formal context, we immediately obtain the two following results:

Lemma 2.1. *Let $\leq_\alpha$ be a partial order over a finite set of size n, and z a variable in $[0,1]$.* $\bigcup_{\leq_A \in \mathcal{LE}(\leq_\alpha)} E_{\leq_A}(z) = E_{\leq_\alpha}(z)$

Lemma 2.2. *Let $\leq_A$ and $\leq_B$ be two different linear extensions in $\mathcal{LE}(\leq_\alpha)$, and μ the Lebesgue measure over $[0,z]^n$. Then $\mu(E_{\leq_B}(z) \cap E_{\leq_A}(z)) = 0$.*

Equipped by this new formal approach, we are now ready to enumerate intricate increasing structures and in particular the one that we are interested in: increasing trees with exactly one repeated label.

3 Partially ordered product

The goal of this section is to be able to count the number of *synchronized trees*, which are labeled trees such that any path from the root to a leaf is strictly increasing, and that there are exactly two nodes with the same label. The number of such objects is counted by the following sequence

$$1, 11, 122, 1518, 21423, 340869, 6058980, 119218860, 2575293165, 60628447215, \dots$$

To be able to enumerate them, we will slightly modify their representation by merging the two nodes with the same label, as can be seen on the example in Figure 1, and define the size to be the number of labels used in order to have the same size for both representations. The DAG representation corresponds to a poset whose transitive relations between elements have been removed.

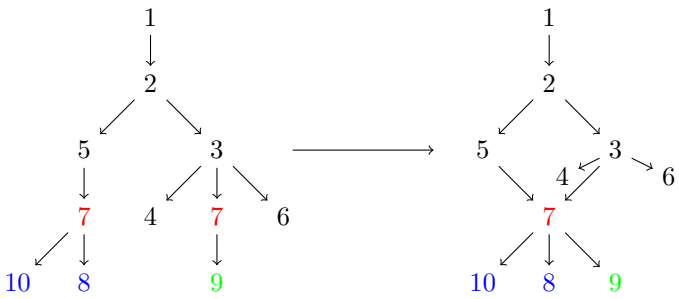

Figure 1: *A synchronized tree*

The binary boxed operator is not enough expressive to specify conveniently the relations between different nodes smaller than a single node (e.g. nodes $3, 5, 7$ in Figure 1). It can be specified by an exhaustive listing of all the possible total orders, but the size of such a listing can be exponential in the poset size.

In order to specify synchronized trees we want to define a partially ordered product of labeled combinatorial classes, which is a product on combinatorial classes based on fixed partial orders containing a smallest element.

Definition 3.1. *Let $\mathcal{A} = \star^{\square}_{\leq_\alpha}(\mathcal{X}_1, \mathcal{X}_2, .., \mathcal{X}_n)$ be the operation which take n labeled classes $\mathcal{X}_1$, $\mathcal{X}_2$, ...,$\mathcal{X}_n$ and a partial order over $\{1, ..., n\}$, and return the subset of the product of those classes, in which for any $i, j \in E$ such that $i \leq_\alpha j$, the smallest label between the $\mathcal{X}_i$ and $\mathcal{X}_j$ components lies in the $\mathcal{X}_i$ component.*

To be consistent with the definition, all the classes must be assumed to have no element of size 0 ($\forall i, X_i(0) = 0$). In the case were $\leq_\alpha = 1 \leq 2$, the classical boxed product defined by [Gre83] can be used:

$$\mathcal{A} = \star^{\square}_{\leq_\alpha}(\mathcal{X}_1, \mathcal{X}_2) = \mathcal{X}_1^{\square} \star \mathcal{X}_2 \qquad \text{and} \qquad A(z) = \int_{t=0}^{z} X_1'(t) X_2(t) dt.$$

For each total order, it is straightforward from there: One can just recursively apply the previous boxed operator to obtain the specification. Another important remark is that any partial order can be decomposed as the sum of all its total orders compatible with it, which gives the following lemma:

Lemma 3.1. $\mathcal{A} = \star^{\square}_{\leq_\alpha}(\mathcal{X}_1, \mathcal{X}_2, .., \mathcal{X}_n) = \sum_{\leq_A \in \mathcal{LE}(\leq_\alpha)} \star^{\square}_{\leq_A}(\mathcal{X}_1, \mathcal{X}_2, .., \mathcal{X}_n)$

The previous lemma allows to analyze the partially ordered set by using the binary boxed operator. However the resulting computation on the generating series is a sum over a possibly exponential number of terms: we want a more concise way to express it. The following theorem show that the partial order we consider for the product, only influences the domain of integration.

Theorem 3.1. *Let $\mathcal{A} = \star^{\square}_{\leq_\alpha}(\mathcal{X}_1, \mathcal{X}_2, .., \mathcal{X}_n)$, and z be smaller than each radius of convergence of the generating functions $X_1', X_2', \ldots, X_n'$:*

$$A(z) = \int_{(z_1,\ldots,z_n) \in E_{\leq_\alpha}(z)} X_1'(z_1) X_2'(z_2) X_3'(z_3) \ldots X_n'(z_n) dz_1 dz_2 dz_3 \ldots dz_n.$$

Proof. By Lemma 3.1, we get

$$A(z) = \sum_{\leq_A \in \mathcal{LE}(\leq_\alpha)} \int_{(z_1,\ldots,z_n) \in E_{\leq_A}(z)} X_1'(z_1) X_2'(z_2) X_3'(z_3) \ldots X_n'(z_n) dz_1 dz_2 dz_3 \ldots dz_n.$$

We can then use Lemma 2.2 to derive

$$A(z) = \int_{\bigcup_{(z_1,\ldots,z_n) \in \leq_A \in \mathcal{LE}(\leq_\alpha)} E_{\leq_A}} X_1'(z_1) X_2'(z_2) X_3'(z_3) \ldots X_n'(z_n) dz_1 dz_2 dz_3 \ldots dz_n,$$

and Lemma 2.1 yields the result. □

This theorem gives a precise description of the value of the generating function, but does not give a straightforward way to calculate the integral, in particular which order of the successive

integrations is the most convenient. A canonical choice for the integrations order consists of having each variable start at 0:

$$A(z) = \int_{z_1=0}^{\min(\{z_i | i \leq_\alpha 1\})} \int_{z_2=0}^{\min(\{z_i | i \leq_\alpha 2\})} \dots \int_{z_n=0}^{\min(\{z_i | i \leq_\alpha n\})} X_1'(z_1)X_2'(z_2)X_3'(z_3)\dots X_n'(z_n)dz_1dz_2dz_3\dots dz_n.$$

In the case of a partial order whose structure is a tree, the `min` operations are always reduced to a single variable, which makes the integration as easy as possible. But in the other cases, one have to find a way to find a good integration order to manage the calculation of the integral. Two techniques can help:

- Change the order of integrations so that the middle values of the order are calculated first, with the consequence that the lower bound of the integrals will no longer be 0:

$$\int_{z_1=0}^{z}\int_{z_2=0}^{z_1}\int_{z_3=0}^{z_2} X_1(z_1)X_2(z_2)X_3(z_3)dz_1dz_2dz_3 = \int_{z_1=0}^{z}\int_{z_3=0}^{z_1} X_1(z_1)X_3(z_3)\int_{z_2=z_3}^{z_1} X_2(z_2)dz_2dz_1dz_3 \tag{1}$$

- Separate in two cases when you have two variables in the domain with the same upper bound, for example a `min` you have a hard time to calculate:

$$\int_{z_1=0}^{z}\int_{z_2=0}^{z} X_1(z_1)X_2(z_2)dz_1dz_2 = \int_{z_1=0}^{z}\int_{z_2=0}^{z_1} X_1(z_1)X_2(z_2)dz_1dz_2 + \int_{z_2=0}^{z}\int_{z_1=0}^{z_2} X_1(z_1)X_2(z_2)dz_1dz_2 \tag{2}$$

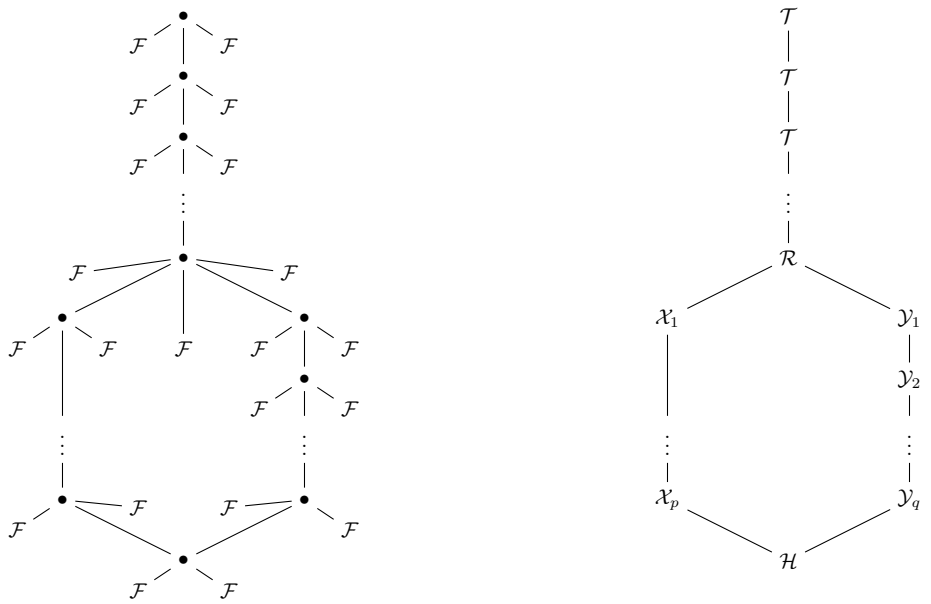

Figure 2: *Combinatorial specification of synchronized trees*

To specify synchronized trees, we note that they are increasing trees, except that there is an unique node where two path separate, to join later. Hence you can model a synchronized tree as an increasing path of nodes with increasing forests, from the class $\mathcal{F}$, grafted to them, which we will call *trunk*, then two distinct increasing branches that eventually merge, with increasing forests grafted at each node (Figure 2). Hence the following system describes the class of synchronized

trees $\mathcal{S}$:

$$\begin{aligned}
\mathcal{S} &= \mathcal{T}^{\square} \star \mathcal{S} + \mathcal{P} \\
\mathcal{P} &= \sum_{p,q\in\mathbb{N}} \star^{\square}_{\leq_{p,q}}(\mathcal{R}, \mathcal{X}_1, ..., \mathcal{X}_p, \mathcal{Y}_1, .., \mathcal{Y}_q, \mathcal{H}) \\
\mathcal{X}_i &= \mathcal{Y}_i = \mathcal{H} = \mathcal{T} = \mathcal{Z}^{\square} \star (\mathcal{F}^2) \\
\mathcal{R} &= \mathcal{Z}^{\square} \star (\mathcal{F}^3)
\end{aligned}$$

With the following notations: $\leq_{p,q}$ is the partial order such that there is a smallest element which corresponds to $\mathcal{R}$, each $\mathcal{X}_i$ (resp. $\mathcal{Y}_i$) is smaller than $\mathcal{X}_{i+1}$ (resp. $\mathcal{Y}_{i+1}$), and there is a maximal element corresponding to $\mathcal{H}$. $\mathcal{F}$ is the class of increasing forest whose exponential generating is the function $F(z) = \frac{1}{\sqrt{1-2z}}$.

Using the Theorem 3.1, we can obtain a generating function for $\mathcal{S}$:

$$\begin{aligned}
S(z) &= \int T'S + P(z) \\
P(z) &= \sum_{p,q\in\mathbb{N}} \int_{r=0}^{z} \int_{x_1=0}^{r} \dots \int_{x_p=0}^{x_{p-1}} \int_{y_1=0}^{r} \dots \int_{y_q=0}^{y_{q-1}} \int_{h=0}^{\min(x_p,y_q)} R'(r)X_1'(x_1)...Y_1'(y_1)H'(h)drdhdx_1 \dots
\end{aligned}$$

By using the technique given in Equation (1), we can write

$$\begin{aligned}
P(z) &= \int_{r=0}^{z} \int_{h=0}^{r} R'(r)H'(h) \left(\sum_{p\in\mathbb{N}} B_p(r,h)\right) \left(\sum_{q\in\mathbb{N}} B_q(r,h)\right) drdh, \\
\text{where} \quad B_p(r,h) &= \int_{t_1=h}^{r} \dots \int_{t_p=h}^{t_{p-1}} T'(t_1)T'(t_2)...T'(t_p).
\end{aligned}$$

Hence:

$$\begin{aligned}
S(z) &= \int_{t=0}^{z} \frac{\sqrt{1-2t}}{\sqrt{1-2z}} P'(t), \\
P(z) &= \int_{r=0}^{z} \int_{h=0}^{r} \frac{1}{(1-2r)\sqrt{1-2r}} \frac{1}{1-2h} \left(\frac{\sqrt{1-2h}}{\sqrt{1-2r}}\right)^2,
\end{aligned}$$

and finally

$$S(z) = \frac{\frac{2z}{1-2\,z} - \log\left(\frac{1}{1-2\,z}\right)}{(4\sqrt{1-2\,z})}.$$

By using the classical transfer theorems [FO90], we get immediately the asymptotic behavior of the number of synchronized trees:

Theorem 3.2. *The number of synchronized trees with n labels, denoted by S_n, satisfies when n tends to infinity*

$$S_n = 2^n\, n! \sqrt{\frac{n}{\pi}} \left(\frac{1}{2} - \frac{\log(n)}{4n}\right) \cdot \left(1 + O\left(\frac{1}{\sqrt{n}}\right)\right).$$

4 Uniform random sampling

In this section we will present a way for generating synchronized trees, but the techniques can apply to other structures with complicated ordered products. It will be based on the classical recursive sampling [FZVC94], but with a loss of efficiency due to the complexity of the problem, as values will have to be calculated on the fly instead of being precalculated.

In order to be able to generate synchronized trees, we must in a first time be able to generate the *shapes* of labeled objects, which are the objects without the labels. The shapes must be sampled with the *weighted distribution*, meaning that if we draw a labeling uniformly among the possible labelings on the generated shape, then we will obtain a uniform labeled object. We must be careful when doing such sampling, as weighted distribution over a shape is in general *not* the uniform generation over all possible shapes. For example, when dealing with general increasing trees, an increasing string of size $n+1$ can have only one labeling, while a root with n children will have $n!$ different labelings, hence the second shape should occur with a probability $n!$ times more often. In the case of increasing trees, in order to sample a shape according to the weighted distribution either one can use the Boltzmann sampler [BRS12] or one can use recursive sampling. However, in our context, we will need further constraint. Once the structure of an increasing tree is sampled, giving an appropriate labeling is also possible, cf. [BGP16].

4.1 Multivariate generating functions

We will extensively use several new formal variables in order to get more precise information about our structures. This technique consists in adding several new variables which will represent other metrics than just size, in order to draw uniformly the values of as many metrics as necessary to reduce our objects into classes of objects that can be simply uniformly generated. Classical examples of such metrics on trees are weights such as the depth, the width or the number of child of the root, ...

If we want to sample uniformly an object from a class with generating function $F(z,u)$, where z carries the size and u another weight, we can first sample the value of the exponent l of u uniformly amongst all the structures, then sample uniformly amongst the object with the measure counted by u equal to l.

Example 4.1. *Let $\mathcal{F}$ be the class of increasing trees. We define $\mathcal{F}$ by the functional equation $\mathcal{F} = \mathcal{Z}^{\square} \star \big(\mathcal{Z} \star \mathcal{Z} \star \mathcal{Z}^{\square} \star (\mathcal{Z} \star \mathcal{Z})\big)$. Let $\Theta\mathcal{F}$ the class of objects of $\mathcal{F}$ where we decide to put a mark on a given node. We can decide that the weight of an object of $\mathcal{F}$ is the number of nodes with a label strictly smaller than the one that is marked. Hence*

$$\theta F(z,u) = \left(\sum_{k=0}^{5} u^k\right) \int_{v=0}^{z} v^2 dv \int_{w=0}^{v} w^2 dw = \left(\sum_{k=0}^{5} u^k\right) \frac{z^6}{18}.$$

And if we want to uniformly sample an element of $\Theta\mathcal{F}$, we can first choose the weight of the element, which, in this case, means choosing uniformly a number between 0 and 5, then uniformly sample a labeling, and finally put the mark on the element corresponding to the weight.

4.2 Synchronized trees

To generate synchronized trees, we will proceed in three steps. Like for many labeled structures, we will sample the shape and then in a second step the labeling. In the case of the synchronized trees, we start by a first additional step that consists in fixing the lengths of the left and right branches, and the length of the trunk.

Length sampling. We will first choose the length of the trunk: We will put a weight on the nodes before the splitting of the trunk.

$$S_1(z,u) = \int_{v=0}^{z} u\frac{\partial T}{\partial v} S_1(v,u)dv + P(z) = \int_{t=0}^{z} \left(\frac{\sqrt{1-2t}}{\sqrt{1-2z}}\right)^u P'(t),$$

$$= -\frac{((u-3)z+1)\sqrt{-2\,z+1}\,(-2\,z+1)^{\frac{1}{2}\,u} - 4\,z^2 + 4\,z - 1}{4\,(u^2-4\,u+3)z^2(-2\,z+1)^{\frac{1}{2}\,u} - 4\,(u^2-4\,u+3)z(-2\,z+1)^{\frac{1}{2}\,u} + (u^2-4\,u+3)(-2\,z+1)^{\frac{1}{2}\,u}}.$$

We sample a given length of the trunk l with probability

$$\mathbb{P}_l = \frac{[u^l z^n]S_1(z,u)}{[z^n]S_1(z,1)}.$$

Then we will choose the size p and q of the left and the right branches respectively (see in Figure 2). For that, we will add two new variables x and y in the generating function of $\mathcal{P}$ to count the size of each respective branch

$$P(z,x,y) = \sum_{p,q\in\mathbb{N}} \int_{r=0}^{z} \int_{x_1=0}^{r} \cdots \int_{y_q=0}^{y_{q-1}} \int_{h=0}^{\min(x_p,y_q)} R'(r)xX_1'(x_1)\ldots xX_p'(x_p)yY_1'(y_1)\ldots H'(h)dr\ldots$$

$$= \int_{r=0}^{z}\int_{h=0}^{r} \frac{1}{(1-2r)\sqrt{1-2r}}\frac{1}{1-2h}\left(\frac{\sqrt{1-2h}}{\sqrt{1-2r}}\right)^{2(x+y)} drdh$$

$$= \frac{\left((x+y)(1-2\,z)^{\frac{1}{2}\,x+\frac{1}{2}\,y+\frac{1}{2}} - (x+y+1)(1-2\,z)^{\frac{1}{2}\,x+\frac{1}{2}\,y} + 1\right)(1-2\,z)^{-\frac{1}{2}\,x-\frac{1}{2}\,y-\frac{1}{2}}}{x^2+(2\,x+1)y+y^2+x}.$$

We can then calculate $S(z,x,y)$ from $P(z,x,y)$, which will depend on the previously sampled length l. Hence for a given n, p and q, the number $n![x^p y^q z^n]S(z,x,y)$ is exactly the number of increasing structures with a left branch of length p and a right branch of length q. We sample a given pair (p,q) with probability

$$\mathbb{P}_{(p,q)} = \frac{[x^p y^q z^n]S(z,x,y)}{[z^n]S(z,1,1)}.$$

Once we have chosen the pair (p,q) with the previous distribution, we can go on with the second step.

Shape sampling. Now we will sample uniformly the shape, which we will do by doing independent calls to samplers for shapes of increasing forests. In order to be able to do that, we need to supply each independent sampler a target size such that the whole shape is uniform. To determine the size of each forest should have, we will use an other set of new variables $a, b_1, b_2, .., b_p, c_1, ..., c_q, f$, and introduce them in the equation so that each forest can have its own variable:

$$P(z,a,b_1,\ldots)$$

$$= \sum_{p,q\in\mathbb{N}} \int_{r=0}^{z} \cdots \int_{h=0}^{\min(x_p,y_q)} aR'(ar)b_1X_1'(b_1x_1)\ldots b_pX_p'(b_px_p)c_1Y_1'(c_1y_1)\ldots dH'(fh)drdhdx_1\ldots$$

Hence for given n, i and j, the value $n![b_i^j z^n]P(z,a,b_1,\ldots)$ is exactly the number of increasing structures with the i-th left branch having a forest of size j. As in the previous step, we will choose a tuple of values according to the repartition induced by $P(z,a,b_1,\ldots)$. Once the size of each component of the shape is determined, we can run an independent sampler for the shapes with the right target size. When we combine all of them, we get a shape who will yield a uniform synchronized tree if uniformly labeled.

Labeling sampling. Here the sampling cannot be obtained by with a Hook-Length formula as we can do for the increasing tree, as it seems that the synchronized node prevent such a formula. We will bypass this problem by splitting our labeling in two distinct parts: The node labels smaller than the synchronized one which will have a weight, and the labels that are larger. To do it, we will once again use multiple variables, one for each forest, then calculate $S(z, x_1, x_2, x_3, \dots, x_p)$, by injecting the bivariate forest built as in example 4.1.

Once the forest size repartitions are known, we sample two independent labeling of increasing trees, one on the subforests (which will have the labels smaller than k), and one on the remaining nodes with labels from $k+1$ to n. The result is an uniformly generated synchronized tree.

5 Perspectives

The combinatorial structure presented above is a single example of an increasing tree structure that gets a useful specification by using Stanley's formal approach. However, the method allows to deal with much more involved structures. In fact, the calculation part associated to the approach relies on successive integrations. Obviously, all properties satisfied by successive integrations are valid in our context of formal (in fact parametrized) integrations. We think for example to linearity, factorizations according to independent parts,...). Thus completely independent parts of the structures can be processed independently form the rest of the structure. For example, in the increasing trees, to each node of the core we attached increasing forests, but we could have attached labeled forest (without order constraint) or even unlabeled forest. In that context, we would use another fresh variable for such substructures and the integration could be processed in two independent steps instead of successive integrations. Another possibility consists in dealing with substructures that can be only enumerated, but without decomposition. Once we have a truncation of its generating series (and eventually an independent random sampler) we can take into account the whole object.

Finally, the approach we developed in this paper gives us a glimpse of completely new combinatorial structures with intricate recursive substructures that can be dealt at once without partitioning entirely the independent structures.

References

[BFS92] F. Bergeron, P. Flajolet, and B. Salvy. Varieties of increasing trees. In *CAAP*, pages 24–48, 1992.

[BGP13] O. Bodini, A. Genitrini, and F. Peschanski. The Combinatorics of Non-determinism. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, volume 24 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 425–436, Guwahati, India, 2013.

[BGP16] O. Bodini, A. Genitrini, and F. Peschanski. A Quantitative Study of Pure Parallel Processes. *Electronic Journal of Combinatorics*, 23(1):P1.11, 39 pages, (electronic), 2016.

[BRS12] O. Bodini, O. Roussel, and M. Soria. Boltzmann samplers for first-order differential specifications. *Discrete Applied Mathematics*, 160(18):2563–2572, December 2012. 15 pages.

[BW91] G. Brightwell and P. Winkler. Counting linear extensions is #P-complete. In *STOC*, pages 175–181, 1991.

[FO90] P. Flajolet and A. M. Odlyzko. Singularity analysis of generating functions. *In SIAM J. Discrete Math.*, 3:216–240, 1990.

[FS09] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[FZVC94] P. Flajolet, P. Zimmermann, and B. Van Cutsem. A Calculus for the Random Generation of Labelled Combinatorial Structures. *Theoretical Computer Science*, 132(1-2):1–35, 1994.

[Gre83] D. H. Greene. *Labelled Formal Languages and Their Uses*. PhD thesis, Stanford, CA, USA, Stanford, CA, USA, 1983. AAI8320712.

[NW78] A. Nijenhuis and H. S. Wilf. *Combinatorial algorithms*. Academic Press, 1978.

[Sta86a] R. P. Stanley. *Enumerative Combinatorics*. Wadsworth & Brooks/Cole, 1986.

[Sta86b] R. P. Stanley. Two poset polytopes. *Discrete & Comp. Geometry*, 1:9–23, 1986.