Exact-size sampling of combinatorial structures in linear time

Konstantinos Panagiotou* Leon Ramzews† and Benedikt Stufler‡

May 3, 2021

Abstract

Various combinatorial classes such as outerplanar graphs and maps, series-parallel graphs, substitution-closed classes of permutations and many more allow an encoding by so-called *enriched trees*, which are rooted Cayley trees with additional structure on the offspring of each node. Under conditions that are satisfied by all aforementioned examples we develop a uniform sampling procedure for enriched trees running in expected linear time. The key ingredient of our sampler is a representation of enriched trees in terms of Galton-Watson trees, for which efficient samplers were developed by Devroye in [19]. Additionally, our method allows us to construct expected linear time samplers for Galton-Watson trees having exactly n (out of $\geq n$ total) nodes with outdegree in some fixed set, implying uniform generation for many combinatorial classes such as dissections of polygons.

1 Introduction and Main Results

Suppose we are given a combinatorial class $\mathcal{C}$, that is, a countable set equipped with a size function $|\cdot| : \mathcal{C} \to \mathbb{N}_0$ such that $\mathcal{C}_n := \{C \in \mathcal{C} : |C| = n\}$ is finite for all $n \in \mathbb{N}_0$. A *sampler* for $\mathcal{C}$ is a set of rules involving random decisions whose output is an element $\mathsf{C} \in \mathcal{C}$ following some given probability distribution. The development of efficient samplers, or equivalently, the efficient random generation of combinatorial objects, is an active and prominent research area with widespread applications. There is a plethora of results and techniques, many of which address specific problems and develop ad hoc methods, and others that create universal techniques that are applicable in various situations.

Let us start right away with an important case that is very well understood and also directly relevant to the results that will be derived here. Let ξ be a random non-negative integer. We create a tree T by starting with a single vertex and attaching to it a number of vertices/children distributed like ξ. Subsequently, we repeat this procedure for every newly created vertex independently. This is the well-known *Galton-Watson* tree with offspring distribution ξ, and by conditioning T to have size n we obtain a random variable T_n taking values in the class of trees with n vertices. For example, if we choose ξ to follow a Poisson distribution with mean one, then the distribution of T_n is just the uniform distribution on the class of all rooted (Cayley) trees with n vertices. In [15] and improving upon many previous results addressing special cases (for example [1, 2] and [14, Ch. XIII.5]) or having a worse running rime [19], Devroye presented a general algorithm for sampling T_n that runs in expected linear time when ξ has finite variance. So, this fundamental case is from today's viewpoint very well understood.

*Department of Mathematics, Ludwigs-Maximilians-Universität München. E-mail: kpanagio@math.lmu.de.

†Department of Mathematics, Ludwigs-Maximilians-Universität München. E-mail: ramzews@math.lmu.de.
Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), Project PA 2080/3-1.

‡Institute for discrete Mathematics and geometry, Technische Universität Wien. E-mail: benedikt.stufler@tuwien.ac.at.

Probably the first systematic approach that applies to a broader setting is the *recursive method* by Nijenhuis and Wilf [29] that can be applied to combinatorial structures defined by specific recursive decompositions. The original method, although quite broad in applicability, was rather inefficient and thus was developed further and improved in several works [22, 13], where eventually samplers with almost linear average time and space complexity were developed. However, all these results are limited to classes that do not allow in general the powerful operation of *substitution*, that is, constructions in which atoms (like vertices or edges in a graph) are replaced by other objects; this limits the applicability of the method to only moderately complex combinatorial classes. Moreover, all variants of the recursive method require (at least) quadratic preprocessing time.

The recursive method is best suited for *exact-size* sampling, where we fix, for example, the size of the object that we want to sample in advance. This paradigm was relaxed in the seminal paper by Duchon, Flajolet, Louchard and Schaeffer [19], allowing samplers to generate objects with varying target size that may be distributed over the whole of $\mathbb{N}$. The so-called *Boltzmann sampling* paradigm developed in that paper is inspired from methods in Physics and postulates to generate objects from the entire class $\mathcal{C}$ with probability proportional to $x^{|C|}/|C|!$ for $C \in \mathcal{C}$, where $x > 0$ is a predefined control parameter; this is a kind of maximum entropy distribution imposed on $\mathcal{C}$. Boltzmann samplers have many advantages: their complexity is (for combinatorial specifications) linear in the size of the generated object, in many cases we have good control of the output size by tuning x, and their description is simple and intuitive. For all these reasons the paper [19] ignited a whole new line of research, where substantial extensions and improvements of Boltzmann samplers were proposed, including the approximate-size linear time sampler for planar graphs [24], Pólya-Boltzmann extensions for unlabelled structures [23, 10], multi-parametric samplers [7, 4], numerical procedures for approximating the values of the generating functions [33] and many more [8, 9, 35].

In this article we combine the world of Devroye – linear time sampling of conditioned Galton-Watson trees – with the world of Boltzmann sampling to assemble efficient linear time samplers that are applicable to a broader spectrum of combinatorial classes. Concretely, the classes that we can treat follow a unified representation in terms of *enriched trees* [32, 37, 38]. Before giving a formal definition later, let us mention a few concrete examples that fall within our scope, for most of which there are no (linear-time) samplers yet.

Example 1.1. *Our approach allows us to treat in a unified setting the following classes:*

1. *connected series-parallel graphs;*
2. *connected outerplanar graphs and (weighted) maps;*
3. *Galton-Watson trees conditioned on the number of vertices whose degree lies in a given set and otherwise no restriction on the size;*
4. *subcritical substitution-closed classes of permutations;*
5. *cographs (with expected runtime being linear in the output size);*
6. *level-k phylogenetic networks.*

The approach taken in this work makes it possible to develop new or to improve all (with the notable exception of outerplanar maps [11]) existing samplers for these examples. For instance, there is no known sampler for series-parallel graphs, and the state-of-the-art sampler for outerplanar graphs [10] runs in expected time $O(n^2)$. Moreover, the sampler for 3. in the example can be used to sample from classes that are in bijection to these objects, for example dissections of polygons. Finally, in [3], among other results, a superlinear uniform sampler for substitution-closed classes

of permutations with a finite number of excluded patterns is presented. Here we will show how to sample from these classes in (optimal) linear time as a consequence of a more general result.

1.1 Main Result: Linear-Time Sampling of Enriched Trees

When measuring the running time/complexity of an algorithm we always assume that we operate under the so-called *RAM model of computation*. This is a widely used approach, followed also in Devroye's paper [15], to establish complexity results that are independent of the actual machine on which the algorithm is executed. In this model we assume that we operate a hypothetical computer called the *Random Access Machine* under the following conditions, see for example [36, Ch. 2]:

1. Basic logical and arithmetic operators like $\{if, call\}$ and $\{+, -, \times, \div\}$ take one time step.
2. Loops and subroutines are not basic operators and count as the composition of many single-step operators.
3. Each memory access takes exactly one step.
4. A number drawn uniformly at random from $(0, 1)$ can be generated in one step.

In particular, in this model real numbers can be stored without loss of precision, whereas in real life one would need to think about the number of decimal places needed so as to not distort the algorithm. In this setting we can already formulate a first consequence of our main result.

Theorem 1.2. *Under the RAM model of computation, the samplers in Section 4 produce objects of size n uniformly at random in expected time $O(n)$ for all the classes in Example 1.1.*

In particular, we develop linear-time algorithms for sampling series-parallel and outerplanar graphs, as well as permutations from specific substitution-closed classes and trees with a given number of leaves.

Towards the proof of Theorem 1.2 we will switch somehow our point of view and look at combinatorial classes as special cases of so-called 'enriched trees'. Generally speaking, the enriched tree viewpoint emphasizes that Galton–Watson trees take a special place among random recursive structures. Specifically, we may sample a random recursive structure by sampling a size-constrained Galton–Watson tree and adding local random 'decorations' later. All classes listed in Example 1.1 are well-known to admit encodings of this form.

Before we present our main result we first need some preparations. Let $\mathcal{R}$ be a combinatorial class. We always consider the labelled setting meaning that all atoms an object of size n is comprised of bear a distinct label, typically in $[n] := \{1, \dots, n\}$. Any other finite set of labels U with $|U| = n$ is also admissible and we write $\mathcal{R}[U]$ to emphasize that we consider objects in $\mathcal{R}_n$ relabelled according to U.

With this notation at hand we may define the main class of objects that we shall study. The class $\mathcal{A}_{\mathcal{R}}$ of *$\mathcal{R}$-enriched trees* contains all rooted labelled unordered trees where the offspring set of each vertex is enriched with an object from $\mathcal{R}$. Let us make this more precise: Denote by $\mathcal{A}$ the class of rooted Cayley trees, i.e., rooted labelled unordered acyclic connected graphs. We (slightly) abuse notation and write $v \in T \in \mathcal{A}$ for the vertex in T bearing the label v. Let P_v be the label set of the offspring of v, where by 'offspring' we define the set of nodes that are connected to v and are at the same time farther away from the root than v. We further define the outdegree $d_T^+(v)$ to be $|P_v|$. Then $\mathcal{A}_{\mathcal{R}}$ contains all sequences of the form

$$(T, (R_v)_{v \in T}), \quad T \in \mathcal{A} \text{ and } R_v \in \mathcal{R}[P_v] \text{ for all } v \in T.$$

As we will see in Section 4, all classes in Example 1.1 (and many more) are (isomorphic to) enriched trees for specific choices of $\mathcal{R}$. Let us write in the sequel $r_k := |\mathcal{R}_k|$ for $k \in \mathbb{N}_0$ and define the generating functions

$$\mathcal{R}(x) := \sum_{k\in\mathbb{N}_0} \frac{r_k}{k!}x^k \quad \text{and} \quad \mathcal{A}_\mathcal{R}(x) := x\mathcal{R}(\mathcal{A}_\mathcal{R}(x)).$$

The function $\mathcal{A}_\mathcal{R}(x)$ is indeed the generating function for $\mathcal{A}_\mathcal{R}$, see e.g. [37]. Let $\rho_\mathcal{R}$ and $\rho_{\mathcal{A}_\mathcal{R}}$ be the radii of convergence of $\mathcal{R}(x)$ and $\mathcal{A}_\mathcal{R}(x)$, respectively. Define, if it exists, the random variable ξ with distribution

$$p_k := \mathbb{P}(\xi = k) := \frac{r_k \mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}})^k}{\mathcal{R}(\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}}))k!}, \qquad k \in \mathbb{N}_0. \tag{1.1}$$

We also need the Boltzmann random variable $\Gamma\mathcal{R}(t)$ for $0 < t < \rho_\mathcal{R}$ given by

$$\mathbb{P}(\Gamma\mathcal{R}(t) = R) = \frac{t^k}{\mathcal{R}(t)k!}, \quad R \in \mathcal{R}_k, k \in \mathbb{N}_0.$$

We now come to the most crucial part, namely the assumptions that we make for the class of enriched trees that we consider.

Definition 1.3. *We call a class of enriches trees* tame *if it has the following properties:*

(A) $\rho_\mathcal{R} > \mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}})$.

(B) $r_0 > 0$ *and there exists* $k \geq 2$ *with* $r_k > 0$.

(C) $\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}})$ *and* $\mathcal{R}(\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}}))$ *are given, and* p_k *can be computed in* $\mathrm{e}^{o(k)}$ *steps for* $k \in \mathbb{N}_0$.

(D) For any $0 < t < \rho_\mathcal{R}$ *there exists a sampling procedure for* $\Gamma\mathcal{R}(t)$ *which runs in expected constant time.*

As we will see, the most critical and essential property is Assumption (A), which ensures that $(p_k)_{k\in\mathbb{N}_0}$ is a probability sequence, that is, $\mathcal{R}(\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}})) < \infty$, and that it is *subcritical*, in the sense that $\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}}) < \rho_\mathcal{R}$; this will imply with well-known results by Janson [25] that p_k has exponential tails. Together with the mild and standard Assumption (B) this will guarantee immediately that a Galton-Watson tree with offspring distribution ξ is non-trivial. Moreover, note that under the RAM model of computation the determination of $\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}})^k/k!$ takes at most k steps, so that (C) actually means that we need to be able to compute r_k (which is in $\mathbb{N}_0$) in $\mathrm{e}^{o(k)}$ steps. Further $\mathcal{A}_\mathcal{R}(\rho_\mathcal{A})$ and $\mathcal{R}(\mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}}))$ being given means that we are able to compute these values beforehand. Finally, (D) is a rather mild condition, since $t < \rho_\mathcal{R}$ and thus all moments of $\Gamma\mathcal{R}(t)$ exist; such samplers can (usually) be designed from the general principles developed in [18, 10].

A milestone towards the proof of Theorem 1.2 is the following fact that we establish in Section 4.

Fact 1.4. *For every combinatorial class* $\mathcal{C}$ *given in Example 1.1 there exists* $\mathcal{R}$ *such that* $\mathcal{C}$ *is isomorphic to* $\mathcal{A}_\mathcal{R}$*, where* $\mathcal{A}_\mathcal{R}$ *is tame.*

In Section 2 we present the backbone of our main result, namely a sampler generating an instance of the random object A_n drawn uniformly at random from all objects in $\mathcal{A}_\mathcal{R}$ of size n. Hence together with the next theorem Fact 1.4 guarantees that the linear time samplers claimed in Theorem 1.2 do indeed exist.

Theorem 1.5. *Assume $\mathcal{A}_\mathcal{R}$ is tame. Then under the RAM model of computation the sampler in Section 2 generates the $\mathcal{R}$-enriched tree A_n in expected time $O(n)$.*

At this point we already anticipate that the algorithm generating A_n is structured as follows. We first sample a Galton-Watson tree with offspring distribution ξ of size n using Devroye's algorithm. Subsequently, for each node we repeatedly call the sampling procedure $\Gamma\mathcal{R}(t)$ until an $\mathcal{R}$-object of the same size as the outdegree of the node at hand is produced. The latter step enhances the offspring of each node with an additional structure leading to an $\mathcal{R}$-enriched tree.

2 The Sampler

In this section we present the sampler for A_n that is needed in the proof of Theorem 1.5. We briefly recall the construction of a Galton-Watson tree with offspring distribution ξ. We start with a distinguished root to which a number of ordered children according to an independent copy of ξ is appended. Repeat this procedure for any node which has not received any children yet or the outcome of the copy of ξ was 0. The result is the arbitrarily sized *unlabelled ordered* rooted tree T such that the distribution of its vertex-degrees is $(p_k)_{k\geq 0}$. The corresponding size-constrained tree is defined as $\mathsf{T}_n := (\mathsf{T} \mid |\mathsf{T}| = n)$ for $n \in \mathbb{N}$. We use the notation that $v \in T$ is some node in the unlabelled ordered tree T. With this at hand, our sampler operates as follows.

Algorithm 2.1. *(Uniform $\mathcal{R}$-enriched tree of size n)*

1. *Generate the Galton-Watson tree T_n with offspring distribution ξ conditioned on having size n as in [15].*
2. *For some $\mathcal{A}(\rho_{\mathcal{A}_\mathcal{R}}) < t_0 < \rho_\mathcal{R}$ repeatedly call for each $v \in \mathsf{T}_n$ the sampler $\Gamma\mathcal{R}(t_0)$ until it produces an object R_v of size $d^+_{\mathsf{T}_n}(v)$.*
3. *Distribute labels in $\{1, \dots, n\}$ uniformly at random and drop the ordering of the vertices afterwards.*

The last step requires some explanation: In Steps 1 and 2 we generate an object $(T, (R_v)_{v\in T})$ where T is an *unlabelled* ordered rooted tree of size n and $R_v \in \mathcal{R}[\{1, \dots, d^+(v)\}]$ for $v \in T$. The ordering of the offspring of $v \in T$ corresponds to a canonical labelling of the vertices, say $(v, 1), \dots, (v, d^+(v))$ so that we may see the object R_v as being labelled with elements from $\{(v, 1), \dots, (v, d^+(v))\}$ (simply consider the bijective mapping $i \mapsto (v, i)$ for $i \in [d^+(v)]$). Naming the root o, the children of the root are hence labelled $(o, 1), \dots, (o, d^+(o))$, the children of the first child of the root by $((o, 1), 1), \dots, ((o, 1), d^+(o, 1))$ and so on. In particular the labels are all distinct and by generating a uniform permutation of $[n]$ we may canonically relabel the entire object with labels in $[n]$.

The next lemma, taken from [38], guarantees that Algorithm 2.1 produces an uniform object of size n from $\mathcal{A}_\mathcal{R}$.

Lemma 2.2 ([38, Lemma 6.1])**.** *In distribution $(\mathsf{T}_n, (\mathsf{R}_v)_{v\in T_n}) = \mathsf{A}_n$.*

So, the proof of Theorem 1.5 boils down to validating that Algorithm 2.1 finishes in linear time. This will be done in Section 3. Before we come to that, let us first have a closer look at the implementation of Step 1 of Algorithm 2.1. Let $\xi_1, \xi_2, \dots$ be iid copies of ξ. Define $S_t :=$

$1+\sum_{1\le i\le t}(\xi_i-1)>0$ for all $1\le t\le n$. Then it is well known that T_n is isomorphic in distribution to

$$(\xi_1,\dots,\xi_n)\ \Big|\ \left\{\sum_{1\le i\le n}\xi_i=n-1, S_t>0 \text{ for all } 1\le t\le n-1 \text{ and } S_n=0\right\}. \tag{2.1}$$

Algorithm 2.3. *([15] Size-constrained Galton-Watson tree T_n with offspring distribution ξ)*

1. *Sample the multinomial random vector $(N_0,N_1,\dots)$ with parameters $(n,p_0,p_1,\dots)$ repeatedly until $\sum_{1\le i\le n} iN_i=n-1$.*
2. *Create a sequence of length n populated with N_j times j for $0\le j\le n$.*
3. *Randomly permute this sequence with each permutation equiprobable.*
4. *Shift the elements of the sequence cyclically until the condition in (2.1) is fulfilled.*

For generating a multinomial vector in the first step [15] proposes a sub-routine depending on binomial random variables.

Algorithm 2.4. *([15] Multinomial random vector $(N_0,N_1,\dots)$ with parameters $(n,p_1,\dots)$)*

1. *Let $N_0=\mathrm{Bin}(n,p_0)$.*
2. *For $i\ge 1$, if $\sum_{0\le j\le i-1}N_j<n$, let $N_i=\mathrm{Bin}(n-\sum_{0\le j\le i-1}N_j, p_i/(1-\sum_{0\le j\le i-1}p_j)$ and otherwise set $N_i=0$.*

Then in [15] it is established that under the RAM model of computation the expected number of steps taken by Algorithm 2.3 is $O(n)$, provided that ξ has finite variance and that it takes one step to generate an independent copy of ξ. Our setting, however, is slightly different as we do not a priory know the entire vector $(p_0,p_1,\dots)$. We need to incorporate the time it takes to compute this vector into the runtime of Step 1 of our Algorithm 2.1. Conveniently, it is sufficient to consider $(p_0,p_1,\dots)$ truncated at $K:=\max_{1\le i\le n}\xi_i$, the step at which $N_j=0$ for all $j>K$ in Algorithm 2.4 and hence the point in time after which p_j for $j>K$ is not needed anymore to finish Step 2. Since ξ has finite exponential moments this will together with (C) turn out to preserve the linear runtime.

Lemma 2.5. *If $\mathcal{A}_{\mathcal{R}}$ is tame, the expected runtime of Algorithm 2.3 is $O(n)$.*

3 Proofs

We first state some facts about the distribution ξ defined in (1.1).

Lemma 3.1. *We have $\mathbb{E}[\xi]=1$ and there exists $\varepsilon>0$ such that $\mathbb{E}[(1+\varepsilon)^{\xi}]<\infty$.*

This is a straightforward consequence of results about simply generated trees in [27]. Due to space restrictions we shift the proof of Lemma 3.1 (which consists basically of adapting the notation in [27] to our setup) to the Appendix. Define the span d by

$$d:=\max\{d\ge 1 : d \mid i \text{ whenever } p_i>0\}.$$

The following local central limit theorem holds for any distribution with finite variance, so in particular in our intended application.

Lemma 3.2 ([27, Lemma 1.4.2]). *Let $\xi_1, \xi_2, \dots$ be iid copies of ξ and set $\Xi_n = \xi_1 + \cdots + \xi_n$. If $\mathbb{E}[\xi] = 1$ and $\sigma^2 < \infty$ it holds that*

$$\mathbb{P}(\Xi_n = n-1) \sim \frac{d}{\sqrt{2\pi\sigma^2 n}}, \quad n \to \infty.$$

The probability that the sum hits a certain value is also maximized at its mean, so that we get (see for example [25, Remark 13.2])

$$\mathbb{P}(\Xi_n = m) \leq \frac{d + o(1)}{\sqrt{2\pi\sigma^2 n}}, \quad m \in \mathbb{N}. \tag{3.1}$$

With these considerations at hand we first prove Lemma 2.5 and then Theorem 1.5. In the following let $\xi_1, \xi_2, \dots$ be independent copies of ξ.

Proof of Lemma 2.5. Define

$$K := \max_{1 \leq i \leq n} \xi_i, \quad \tau_n := \mathbb{E}[K] \quad \text{and} \quad \varphi_n := \mathbb{P}\Big(\sum_{1 \leq i \leq n} \xi_i = n-1 \Big).$$

According to [15] the expected time until Algorithm 2.4 finishes is $O\big((1+\tau_n)/\varphi_n\big) + O(n)$ if the probabilities in $(p_0, p_1, \dots)$ are given. This runtime is split up into $1 + \tau_n$ which is the expected time until a multinomial vector $(N_0, N_1, \dots, N_K, 0, \dots)$ in Step 1 is drawn and it takes on average φ_n^{-1} rejections until a vector is found such that $\sum_{1 \leq i \leq n} i N_i = n-1$. The expected time for Steps 2-4 is $O(n)$.

In our setting we additionally have to take care of the computation time of $(p_0, p_1, \dots, p_K)$ in Step 1. Assumption (C) yields that there exists a non-negative sequence $(f_i)_{i \in \mathbb{N}_0}$ with $f_n \to 0$ as $n \to \infty$ such that the expected time to compute $(p_0, p_1, \dots, p_K)$ is

$$\mathbb{E}\Big[\sum_{0 \leq i \leq K} \mathrm{e}^{f_i \cdot i} \Big] = \sum_{m \geq 1} \mathbb{P}(K = m) \sum_{0 \leq i \leq m} \mathrm{e}^{f_i \cdot i}. \tag{3.2}$$

Let $\varepsilon' > 0$ be fixed but arbitrary. Then there exists $L \in \mathbb{N}$ such that $f_i \leq \varepsilon'$ for all $i \geq L$. Hence, if $m \geq L$ there exists a constant $C' > 0$

$$\sum_{0 \leq i \leq m} \mathrm{e}^{f_i \cdot i} = \sum_{0 \leq i \leq L} \mathrm{e}^{f_i \cdot i} + \sum_{L \leq i \leq m} \mathrm{e}^{f_i \cdot i} \leq C' + (m - L)\mathrm{e}^{\varepsilon' m}.$$

If $m < L$ the expression above is simply $\leq C'$. In any case we obtain constants $C > 0$ and $\varepsilon > 0$ such that uniformly in m

$$\sum_{0 \leq i \leq m} \mathrm{e}^{f_i \cdot i} \leq C \cdot \mathrm{e}^{\varepsilon m}.$$

Note that since ε' is arbitrarily small (by choosing a larger L) the parameter ε can be chosen as small as fits our needs as well. We proceed with the computation of (3.2) by the estimate

$$\mathbb{E}\Big[\sum_{0 \leq i \leq K} \mathrm{e}^{f_i \cdot i} \Big] \leq C \sum_{m \geq 1} \mathbb{P}(K = m)\, \mathrm{e}^{\varepsilon m} = C\mathbb{E}[\mathrm{e}^{\varepsilon K}].$$

Let $t > 0$ be such that $\mathbb{E}[\mathrm{e}^{t\xi}] < \infty$. The existence of such t is guaranteed by the finite exponential moments of ξ stated in Lemma 3.1. We use the Log-Sum-Exp function estimate $\max_{1 \leq i \leq n} x_i \leq \log(\mathrm{e}^{\alpha x_1} + \cdots + \mathrm{e}^{\alpha x_n})/\alpha$ for any $(x_1, \dots, x_n) \in \mathbb{N}_0^n$ and $\alpha > 0$ to obtain the upper bound

$$\mathbb{E}[\mathrm{e}^{\varepsilon K}] \leq \mathbb{E}[\mathrm{e}^{\varepsilon/t \cdot \log(\exp(t\xi_1) + \cdots + \exp(t\xi_n))}].$$

We pick ε such that $\varepsilon/t < 1/2$. Then the Jensen inequality entails

$$\mathbb{E}[\mathrm{e}^{\varepsilon/t\cdot\log(\exp(t\xi_1)+\cdots+\exp(t\xi_n))}] \le n^{\varepsilon/t}\mathbb{E}[\mathrm{e}^{t\xi}]^{\varepsilon/t} = O(n^{1/2}).$$

Hence the waiting time adapted to our setting is bounded from above by a constant times

$$\frac{1+\tau_n+O(n^{1/2})}{\varphi_n}.$$

Since in particular the finite exponential moments imply $\mathbb{E}[\xi^2] < \infty$ we get due to [15] that $\tau_n = O(n^{1/2})$ and $\varphi_n^{-1} = O(n^{1/2})$. All in all this shows that Step 1 is completed in $O(n)$ steps. The remaining steps stay unchanged, so that the proof is finished. □

Proof of Theorem 1.5. Subsequently we go through each step in Algorithm 2.1 and explain how the expected runtime of $O(n)$ is achieved. For Step 1 see Lemma 2.5. Let us next investigate Step 2. Let $\mathcal{A}(\rho_\mathcal{A}) < t_0 < \rho_\mathcal{R}$. The probability that $\Gamma\mathcal{R}(t_0)$ produces a k-sized object is given by

$$\mathbb{P}(|\Gamma\mathcal{R}(t_0)| = k) = \frac{r_k t_0^k}{k!\mathcal{R}(t_0)}, \qquad k \in \mathbb{N}.$$

Since $\Gamma\mathcal{R}(t_0)$ runs in expected constant time, the expected time until an object of size $k \in \mathbb{N}_0$ is drawn is computed via the geometric waiting time

$$f(k) = \frac{r_k t_0^k}{k!\mathcal{R}(t_0)} \sum_{\ell\ge 1} \ell \left(1 - \frac{r_k t_0^k}{k!\mathcal{R}(t_0)}\right)^{\ell-1} = \frac{k!\mathcal{R}(t_0)}{r_k t_0^k}.$$

Let W_n denote the total waiting time until Step 2 is completed and set $\Xi_n = \sum_{1\le i\le n}\xi_i$. Then using the equivalent representation of T_n in (2.1) we deduce

$$\mathbb{E}[W_n] = \mathbb{E}\left[\sum_{v\in T_n} f\left(d^+_{T_n}(v)\right)\right] = \mathbb{E}\left[\sum_{1\le i\le n} f(\xi_i) \;\middle|\; \Xi_n = n-1\right] = n\mathbb{E}\left[f(\xi_1) \mid \Xi_n = n-1\right].$$

The condition in (2.1) involving S_t can be omitted in the conditional expectation as it does not influence the value of the sum over $f(\xi_i)$, $1 \le i \le n$. Due to Lemma 3.1 we obtain with Lemma 3.2 and Equation (3.1)

$$\begin{aligned}\mathbb{E}\left[f(\xi_1) \mid \Xi_n = n-1\right] &= \sum_{k\ge 1} f(k)\mathbb{P}(\xi = k)\frac{\mathbb{P}(\Xi_{n-1} = n-k-1)}{\mathbb{P}(\Xi_n = n-1)} \\ &= O\left(\sum_{k\ge 1}\frac{\mathcal{R}(t_0)}{\mathcal{A}(\rho_\mathcal{A})}\left(\frac{\mathcal{A}(\rho_\mathcal{A})}{t_0}\right)^k\right).\end{aligned}$$

Since $t_0 > \mathcal{A}(\rho_\mathcal{A})$ the previous expression is $O(1)$. This immediately gives us $\mathbb{E}[W_n] = O(n)$ as desired. Finally, generating a uniform permutation of $[n]$ in Step 3 takes $O(n)$ steps, see for example [26, Alg. P (Shuffling)]. □

4 Applications

Due to space limitations we only present an application of our results to the class of series-parallel graphs (which is a special case of subcritical classes of connected graphs). The other examples will be treated in detail in a future journal version of this work. Nonetheless, it is far from obvious how to assemble from Algorithm 2.1 a sampler for Galton-Watson trees conditioned on the number of nodes whose outdegree lies in a given set. Thus, for completeness, we devoted Section C in the Appendix to this problem.

4.1 Subcritical classes of connected graphs

Let us first recall a few basic notions from graph theory. A subgraph of a graph G is called a *block* of G if is is maximal such that it is either (isomorphic) to an edge or if it is 2-connected otherwise. Then we call a class $\mathcal{G}$ *block-stable* if it contains the graph that is isomorphic to a single edge and that has the property that a graph belongs to $\mathcal{G}$ if and only if all of its blocks belong to $\mathcal{G}$. Block stable classes are ubiquitous, and include, for example, classes that are specified by excluding a finite list of minors.

It is well-known, see for example [12], that any block stable class $\mathcal{C}$ of connected graphs satisfies the relation

$$\mathcal{C}^\bullet \simeq \mathcal{X} \cdot (\mathrm{SET} \circ \mathcal{B}' \circ \mathcal{C}^\bullet),$$

where $\mathcal{B}$ denotes the class of 2-connected graphs in $\mathcal{C}$ together with the graph that is isomorphic to an edge and $\mathcal{X}$ contains a single vertex. Here, the combinatorial construction '$\cdot$' means forming the Cartesian product, '$\circ$' stands for substitution at nodes and 'SET' for forming a set of objects. The pointed class $\mathcal{C}^\bullet$ contains all objects in $\mathcal{C}$ together with one distinguished vertex and the derived class $\mathcal{B}'$ contains all objects in $\mathcal{B}$ with one of the vertices being replaced by a special placeholder-atom neither bearing a label nor contributing to the size. For a thorough treatment of such combinatorial constructions we refer to [21, 5]; moreover, we have included for completeness a brief but formal description in the Appendix. This decomposition of connected graphs follows directly from the decomposition into blocks, that enables us to describe a graph in terms of a tree and associated subgraphs of higher connectivity. What is important here is that block stable classes are isomorphic to enriched trees $\mathcal{A}_\mathcal{R}$ with $\mathcal{R} = \mathrm{SET} \circ \mathcal{B}'$; this observation is not new and was exploited also elsewhere [31, 37, 38].

From the description of block stable classes of connected graphs we immediately obtain the relation

$$C'(x) = \exp(B'(xC'(x)))$$

for the corresponding exponential generating functions. We will study here the particular case in which the composition of the generating functions is *subcritical*, meaning that the largest value that $xC'(x)$ can attain, where x is at most the radius of convergence of C, lies strictly within the disc of convergence of B'.

Definition 4.1. *Let $\mathcal{C}$ be a block stable class of connected graphs and $\mathcal{B}$ the corresponding class of 2-connected graphs together with the graph that is isomorphic to an edge. Let $\rho_\mathcal{C}$ and $\rho_\mathcal{B}$ denote the radii of convergence of $C(x)$ and $B(x)$. We say that $\mathcal{C}$ is* subcritical *if $\rho_\mathcal{B} > \rho_\mathcal{C} C'(\rho_\mathcal{C})$.*

Subcritical graph classes have been studied from various viewpoints and include many important classes like trees, outerplanar and series-parallel graphs, but exclude also others, like planar graphs. From an *analytical* viewpoint, in the subcritical case the behavior of C near its singular points is not dictated by the behavior of B, but it is rather a consequence of the composition $B'(xC'(x))$; this is in stark contrast to critical compositions (where necessarily $\rho_\mathcal{B} = \rho_\mathcal{C} C'(\rho_\mathcal{C})$), where there is an explicit interplay. From a *combinatorial* viewpoint subcritical classes are very much tree-like, in the sense that the blocks are typically small, the largest one having at most logarithmic size in the size of the graph. This makes it possible to study subcritical classes rather abstractly without explicitly fixing $\mathcal{B}$, and by now there are many results that address the local as well the global structure of 'typical' members of such classes [16, 6, 17, 30]. Here we extend this list of results by providing samplers for outerplanar and series-parallel graphs; many other classes for which there is a decomposition of the 2-connected graphs can be treated analogously.

In order to apply our main fact we begin with the following fact, which is a simple consequence of the definition of subcritical classes.

Fact 4.2. *Let $\mathcal{C}$ be subcritical. Then $\mathcal{C}$ has the properties (A) and (B) in the definition of a tame class of enriched trees.*

So, what remains to verify for the classes at hand are the properties (C) and (D) in the definition of tame enriched trees. In particular, we have to show that $(\mathrm{SET} \circ \mathcal{B}')_k$ can be computed in time $e^{o(k)}$ and that we have a Boltzmann sampler for $(\mathrm{SET} \circ \mathcal{B}')$. We show this for the particular case of series-parallel graphs, as they have the most complex description; in all other cases (for example, outerplanar graphs) we proceed qualitatively similar, albeit not as complex. Due to space limitations we restrict the presentation here to series-parallel graphs.

Series-Parallel Graphs In the case of series-parallel graphs we will use the following property of the associated class $\mathcal{B}'$, see for example [6, 12].

Lemma 4.3. *Let $\mathcal{X}$ be the class of graphs consisting of a single graph that is an isolated vertex, $\mathcal{X}_2$ the class consisting of a single graph that contains exactly two isolated vertices, and e the class consisting of a single graph that is an isolated edge, where the size of it is defined to be zero. Then the class $\mathcal{B}'$ has the decomposition*

$$\mathcal{B}' + \mathcal{B}'_{(rm)} \simeq e \times \mathcal{X} + \mathcal{B}'_{(r)} + \mathcal{B}'_{(m)},$$

where $\mathcal{B}'_{(rm)} \subseteq \mathcal{B}'$ and

$$\mathcal{B}'_{(r)} \simeq \mathcal{X}_2 \cdot (e + \mathcal{P})^2 \cdot \mathcal{D}, \;\; \mathcal{B}'_{(m)} \simeq \mathcal{X} \cdot \big(e \cdot \mathrm{SET}_{\geq 2}(\mathcal{S}) + \mathrm{SET}_{\geq 3}(\mathcal{S})\big), \;\; \mathcal{B}'_{(rm)} \simeq \mathcal{X} \cdot \mathcal{P} \cdot \mathcal{S},$$

and

$$\mathcal{D} \simeq e + \mathcal{S} + \mathcal{P}, \mathcal{S} = (e + \mathcal{P}) \cdot \mathcal{X} \cdot \mathcal{D}, \mathcal{P} = e \cdot \mathrm{SET}_{\geq 1}(\mathcal{S}) + \mathrm{SET}_{\geq 2}(\mathcal{S}).$$

With this lemma at hand it is straightforward to compute determine $\mathcal{R}_k = (\mathrm{SET} \circ \mathcal{B}')_k$ in $e^{o(k)}$ steps; this can be actually performed in an almost linear number of steps by the combinatorial Newton iteration in [33]. Moreover, we can derive a Boltzmann sampler for $\mathcal{B}' + \mathcal{B}'_{(rm)}$ from the general principles for the construction of Boltzmann samplers from [19]. In addition to that, since $\mathcal{B}'_{(rm)} \subseteq \mathcal{B}'$, we obtain a sampler for $\mathcal{B}'$ by rejection: if the sampled graph is in $\mathcal{B}'_{(rm)}$, we reject it with probability $1/2$ and repeat the experiment, see also [24] for related constructions. Finally, constructing a sampler for $\mathcal{R} = \mathrm{SET} \circ \mathcal{B}'$ can also be done by using the general principles in [19].

References

[1] D. B. Arnold and M. R. Sleep. Uniform random generation of balanced parenthesis strings. *ACM Trans. Program. Lang. Syst.*, 2(1):122–128, Jan. 1980. ISSN 0164-0925. doi: 10.1145/357084.357091. URL https://doi.org/10.1145/357084.357091.

[2] A. Bacher, O. Bodini, and A. Jacquot. Exact-size sampling for Motzkin trees in linear time via Boltzmann samplers and holonomic specification. In *ANALCO13—Meeting on Analytic Algorithmics and Combinatorics*, pages 52–61. SIAM, Philadelphia, PA, 2013. doi: 10.1137/1.9781611973037.7. URL https://doi.org/10.1137/1.9781611973037.7.

[3] F. Bassino, M. Bouvel, A. Pierrot, C. Pivoteau, and D. Rossin. An algorithm computing combinatorial specifications of permutation classes. *Discrete Appl. Math.*, 224:16–44, 2017. ISSN 0166-218X. doi: 10.1016/j.dam.2017.02.013. URL https://doi.org/10.1016/j.dam.2017.02.013.

[4] M. Bendkowski, O. Bodini, and S. Dovgal. Polynomial tuning of multiparametric combinatorial samplers. In *2018 Proceedings of the Fifteenth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 92–106. SIAM, Philadelphia, PA, 2018. doi: 10.1137/1.9781611975062.9. URL https://doi.org/10.1137/1.9781611975062.9.

[5] F. Bergeron, G. Labelle, and P. Leroux. *Combinatorial species and tree-like structures*, volume 67 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1998. ISBN 0-521-57323-8. Translated from the 1994 French original by Margaret Readdy, With a foreword by Gian-Carlo Rota.

[6] N. Bernasconi, K. Panagiotou, and A. Steger. The degree sequence of random graphs from subcritical classes. *Combin. Probab. Comput.*, 18(5):647–681, 2009. ISSN 0963-5483. doi: 10.1017/S0963548309990368. URL http://dx.doi.org/10.1017/S0963548309990368.

[7] O. Bodini and Y. Ponty. Multi-dimensional Boltzmann sampling of languages. In *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'10)*, Discrete Math. Theor. Comput. Sci. Proc., AM, pages 49–63. Assoc. Discrete Math. Theor. Comput. Sci., Nancy, 2010.

[8] O. Bodini, E. Fusy, and C. Pivoteau. Random sampling of plane partitions. *Combin. Probab. Comput.*, 19(2):201–226, 2010. ISSN 0963-5483. doi: 10.1017/S0963548309990332. URL https://doi.org/10.1017/S0963548309990332.

[9] O. Bodini, D. Gardy, and A. Jacquot. Asymptotics and random sampling for BCI and BCK lambda terms. *Theoret. Comput. Sci.*, 502:227–238, 2013. ISSN 0304-3975. doi: 10.1016/j.tcs.2013.01.008. URL https://doi.org/10.1016/j.tcs.2013.01.008.

[10] M. Bodirsky, É. Fusy, M. Kang, and S. Vigerske. Boltzmann samplers, Pólya theory, and cycle pointing. *SIAM J. Comput.*, 40(3):721–769, 2011. ISSN 0097-5397. doi: 10.1137/100790082. URL http://dx.doi.org/10.1137/100790082.

[11] N. Bonichon, C. Gavoille, and N. Hanusse. Canonical decomposition of outerplanar maps and application to enumeration, coding and generation. *J. Graph Algorithms Appl.*, 9(2):185–204 (electronic), 2005. ISSN 1526-1719. doi: 10.7155/jgaa.00105. URL http://dx.doi.org/10.7155/jgaa.00105.

[12] G. Chapuy, E. Fusy, M. Kang, and B. Shoilekova. A complete grammar for decomposing a family of graphs into 3-connected components. *Electron. J. Combin.*, 15(1):Research Paper 148, 39, 2008. URL http://www.combinatorics.org/Volume_15/Abstracts/v15i1r148.html.

[13] A. Denise and P. Zimmermann. Uniform random generation of decomposable structures using floating-point arithmetic. *Theoretical Computer Science*, 218(2):233–248, 1999.

[14] L. Devroye. *Non-Uniform Random Variate Generation*. Springer New York, 1986. ISBN 9783540963059. URL https://books.google.de/books?id=mEw_AQAAIAAJ.

[15] L. Devroye. Simulating size-constrained Galton-Watson trees. *SIAM J. Comput.*, 41(1):1–11, 2012. ISSN 0097-5397. doi: 10.1137/090766632. URL https://doi.org/10.1137/090766632.

[16] M. Drmota and M. Noy. Extremal parameters in sub-critical graph classes. In *ANALCO13—Meeting on Analytic Algorithmics and Combinatorics*, pages 1–7. SIAM, Philadelphia, PA, 2013. doi: 10.1137/1.9781611973037.1. URL http://dx.doi.org/10.1137/1.9781611973037.1.

[17] M. Drmota, É. Fusy, M. Kang, V. Kraus, and J. Rué. Asymptotic study of subcritical graph classes. *SIAM J. Discrete Math.*, 25(4):1615–1651, 2011. ISSN 0895-4801. doi: 10.1137/100790161. URL http://dx.doi.org/10.1137/100790161.

[18] P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Random sampling from Boltzmann principles. In *Automata, languages and programming*, volume 2380 of *Lecture Notes in Comput. Sci.*, pages 501–513. Springer, Berlin, 2002. doi: 10.1007/3-540-45465-9_43. URL http://dx.doi.org/10.1007/3-540-45465-9_43.

[19] P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combin. Probab. Comput.*, 13(4-5):577–625, 2004. ISSN 0963-5483. doi: 10.1017/S0963548304006315. URL http://dx.doi.org/10.1017/S0963548304006315.

[20] R. Ehrenborg and M. Méndez. Schröder parenthesizations and chordates. *J. Combin. Theory Ser. A*, 67(2):127–139, 1994. ISSN 0097-3165. doi: 10.1016/0097-3165(94)90008-6. URL http://dx.doi.org/10.1016/0097-3165(94)90008-6.

[21] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. Cambridge University Press, Cambridge, 2009. ISBN 978-0-521-89806-5. doi: 10.1017/CBO9780511801655. URL http://dx.doi.org/10.1017/CBO9780511801655.

[22] P. Flajolet, P. Zimmerman, and B. Van Cutsem. A calculus for the random generation of labelled combinatorial structures. *Theoret. Comput. Sci.*, 132(1-2):1–35, 1994. ISSN 0304-3975. doi: 10.1016/0304-3975(94)90226-7. URL https://doi.org/10.1016/0304-3975(94)90226-7.

[23] P. Flajolet, É. Fusy, and C. Pivoteau. Boltzmann sampling of unlabelled structures. In *Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments and the Fourth Workshop on Analytic Algorithmics and Combinatorics*, pages 201–211. SIAM, Philadelphia, PA, 2007.

[24] E. Fusy. Uniform random sampling of planar graphs in linear time. *Random Structures Algorithms*, 35(4):464–522, 2009. ISSN 1042-9832. doi: 10.1002/rsa.20275. URL https://doi.org/10.1002/rsa.20275.

[25] S. Janson. Simply generated trees, conditioned Galton-Watson trees, random allocations and condensation. *Probab. Surv.*, 9:103–252, 2012. ISSN 1549-5787. doi: 10.1214/11-PS188. URL http://dx.doi.org/10.1214/11-PS188.

[26] D. E. Knuth. *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. Addison Wesley Longman Publishing Co., Inc., USA, 1997. ISBN 0201896834.

[27] V. F. Kolchin. *Random mappings*. Translation Series in Mathematics and Engineering. Optimization Software, Inc., Publications Division, New York, 1986. ISBN 0-911575-16-2. Translated from the Russian, With a foreword by S. R. S. Varadhan.

[28] I. Kortchemski. Invariance principles for Galton-Watson trees conditioned on the number of leaves. *Stochastic Process. Appl.*, 122(9):3126–3172, 2012. ISSN 0304-4149. doi: 10.1016/j.spa.2012.05.013. URL http://dx.doi.org/10.1016/j.spa.2012.05.013.

[29] A. Nijenhuis and H. S. Wilf. *Combinatorial algorithms*. Academic Press, Inc. [Harcourt Brace Jovanovich, Publishers], New York-London, second edition, 1978. ISBN 0-12-519260-6. For computers and calculators, Computer Science and Applied Mathematics.

[30] K. Panagiotou and A. Steger. Maximal biconnected subgraphs of random planar graphs. *ACM Trans. Algorithms*, 6(2):Art. 31, 21, 2010. ISSN 1549-6325. doi: 10.1145/1721837.1721847. URL http://dx.doi.org/10.1145/1721837.1721847.

[31] K. Panagiotou, B. Stufler, and K. Weller. Scaling limits of random graphs from subcritical classes. *Ann. Probab.*, 44(5):3291–3334, 2016. ISSN 0091-1798. doi: 10.1214/15-AOP1048. URL https://doi.org/10.1214/15-AOP1048.

[32] K. Panagiotou, B. Stufler, and K. Weller. Scaling limits of random graphs from subcritical classes. *Ann. Probab.*, 44(5):3291–3334, 2016. ISSN 0091-1798. doi: 10.1214/15-AOP1048.

[33] C. Pivoteau, B. Salvy, and M. Soria. Algorithms for combinatorial structures: well-founded systems and Newton iterations. *J. Combin. Theory Ser. A*, 119(8):1711–1773, 2012. ISSN 0097-3165. doi: 10.1016/j.jcta.2012.05.007. URL https://doi.org/10.1016/j.jcta.2012.05.007.

[34] D. Rizzolo. Scaling limits of Markov branching trees and Galton-Watson trees conditioned on the number of vertices with out-degree in a given set. *Ann. Inst. Henri Poincaré Probab. Stat.*, 51(2):512–532, 2015. ISSN 0246-0203. URL https://doi.org/10.1214/13-AIHP594.

[35] O. Roussel and M. Soria. Boltzmann sampling of ordered structures. In *LAGOS'09—V Latin-American Algorithms, Graphs and Optimization Symposium*, volume 35 of *Electron. Notes Discrete Math.*, pages 305–310. Elsevier Sci. B. V., Amsterdam, 2009. doi: 10.1016/j.endm.2009.11.050. URL https://doi.org/10.1016/j.endm.2009.11.050.

[36] S. S. Skiena. *The Algorithm Design Manual*. Texts in Computer Science. Springer International Publishing, London, 3 edition, 2020. ISBN 978-3-030-54255-9. doi: 10.1007/978-3-030-54256-6.

[37] B. Stufler. Random enriched trees with applications to random graphs. *Electronic Journal of Combinatorics*, 25(3), 2018.

[38] B. Stufler. Limits of random tree-like discrete structures. *Probab. Surv.*, 17:318–477, 2020. doi: 10.1214/19-PS338. URL https://doi.org/10.1214/19-PS338.

A Combinatorial Classes and $\mathcal{R}$-enriched trees

A.1 Combinatorial Classes

In this section we recall some background information about combinatorial classes. A comprehensive survey of the theory is given in the books [21] and [5]. A (combinatorial) class is given by a countable set $\mathcal{C}$ equipped with a size function $|\cdot| : \mathcal{C} \to \mathbb{N}_0$ such that $\mathcal{C}_n := |\{C \in \mathcal{C} : |C| = n\}|$ is finite for all $n \in \mathbb{N}_0$. Elements of $\mathcal{C}$ are called objects or structures and any object in $\mathcal{C}_n$ is said to be comprised of n atoms or to be of size n. We call $\mathcal{C}$ labelled if each atom of an object in $\mathcal{C}$ bears a distinct label. For convenience, let the label set of any $C \in \mathcal{C}_n$ be given by $\{1, \dots, n\}$. If we want to stress out a different labelling, we simply write $C \in \mathcal{C}[U]$ describing an object $C \in \mathcal{C}_{|U|}$ labelled according to the finite set U. For any bijection $\sigma : U \to V$ between finite sets U, V and $C \in \mathcal{C}[U]$ we write $\sigma.C$

for the object obtained by replacing the label $\ell_v \in U$ of each atom v of C by $\sigma(\ell_v) \in V$. Clearly the resulting object is in $\mathcal{C}[V]$. For coherence we assume that $C \in \mathcal{C}[U]$ implies that $\sigma.C \in \mathcal{C}[U]$ for any bijection $\sigma : U \to U$. The (exponential) generating series of $\mathcal{C}$ is the formal power series defined by

$$\mathcal{C}(x) := \sum_{n \geq 0} |\mathcal{C}_n| \frac{x^n}{n!}.$$

Example A.1. *Consider the class $\mathcal{A}$ of rooted labelled unordered acyclic connected graphs, in short Cayley trees. In the graph setting atoms are called vertices or nodes. For example we see in Figure 1 some $T \in \mathcal{A}_5$ with labels in $\{1, \ldots, 5\}$. Applying the bijection σ mapping 1 to a, 2 to b and so on, we retrieve an object $\sigma.T$ in $\mathcal{A}[\{a, b, c, d, e\}]$.*

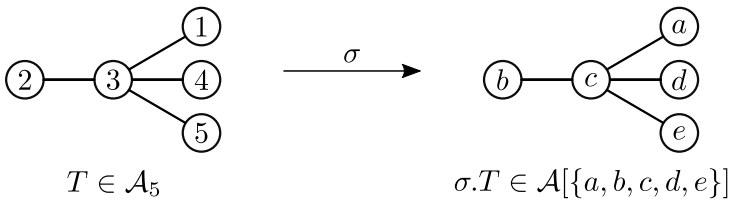

Figure 1: Relabelling T under the bijection σ.

A.1.1 Basic Classes

The basic combinatorial classes are the empty class $\emptyset$, the atomic class $\mathcal{X}$, the set class SET, the cycle class CYC and the sequence class SEQ. Let $\mathcal{S}_n$ be the symmetric group containing all permutations of $[n]$ and write $(a_1, \ldots, a_n) \simeq (b_1, \ldots, b_n)$ if one of the two sequences is obtained by cyclically shifting the indices of the other. Then the basic classes are defined by, letting $n \in \mathbb{N}_0$,

- $|\emptyset_n| = \mathbb{1}_{n=0}$,
- $|\mathcal{X}_n| = \mathbb{1}_{n=1}$,
- $\mathrm{SET}_n = \{\{1, \ldots, n\}\}$,
- $\mathrm{CYC}_n = \{(\sigma(1), \ldots, \sigma(n))_{\simeq} : \sigma \in \mathcal{S}_n\}$ and
- $\mathrm{SEQ}_n = \{(\sigma(1), \ldots, \sigma(n)) : \sigma \in \mathcal{S}_n\}$.

The respective generating series are computed to be

$$\emptyset(x) = 1, \quad \mathcal{X}(x) = x, \quad \mathrm{SET}(x) = \exp(x), \quad \mathrm{CYC}(x) = \log \frac{1}{1-x} \quad \text{and} \quad \mathrm{SEQ}(x) = \frac{1}{1-x}.$$

A.1.2 Constructions

Given classes $\mathcal{A}$ and $\mathcal{B}$ there are several ways to construct more complex classes.

Pointing The collection of elements (A, a) where $A \in \mathcal{A}$ and a is a distinguished atom (the root) in A forms the pointed class $\mathcal{A}^\bullet$. Since $|\mathcal{A}^\bullet_n| = n|\mathcal{A}_n|$ we obtain

$$\mathcal{A}^\bullet(x) = x \frac{\partial}{\partial x} A(x).$$

Derivation Let $\star$ denote a special label such that any atom bearing this label does not contribute to the total size of the object at hand. The derived class $\mathcal{A}'$ is given by the collection of objects in $\mathcal{A}$ where the largest label is replaced by $\star$, i.e. $\mathcal{A}'_{n-1} := \mathcal{A}[\{1, \dots, n-1, \star\}]$ for all $n \in \mathbb{N}$. Hence the generating series is computed to be

$$\mathcal{A}'(x) = \frac{\partial}{\partial x}\mathcal{A}(x).$$

Product The product class $\mathcal{A} \cdot \mathcal{B}$ contains all tuples (A, B) with $A \in \mathcal{A}$ and $B \in \mathcal{B}$ relabelled with labels in $\{1, \dots, |A| + |B|\}$. The size function is defined by $|(A, B)| = |A| + |B|$. The generating series is

$$(\mathcal{A} \cdot \mathcal{B})(x) = \mathcal{A}(x)\mathcal{B}(x).$$

Substitution For this construction we assume $\mathcal{B}_0 = \emptyset$. An object in the substitution class $\mathcal{A} \circ \mathcal{B}$ is comprised of an $\mathcal{A}$-object whose atoms are replaced by $\mathcal{B}$-objects. In other words, $\mathcal{A} \circ \mathcal{B}$ contains all equivalence classes of sequences of the form

$$(A, B_1, \dots, B_k)_{\simeq}, \quad A \in \mathcal{A}[\{P_1, \dots, P_k\}], B_i \in \mathcal{B}[P_i], 1 \leq i \leq k,$$

where $k \geq 0$ and $P_1, \dots, P_k$ is a partition of $\{1, \dots, \sum_{1 \leq i \leq k} |B_i|\}$ with $|P_i| = |B_i|$ for $1 \leq i \leq k$. The equivalence relation "$\simeq$" terms two sequences $(A, B_1, \dots, B_k)$ and $(A', B'_1, \dots, B'_k)$ isomorphic if $A = A'$ and for any permutation $\sigma : \{1, \dots, k\} \to \{1, \dots, k\}$ such that $\sigma.A = A$ it holds that $B_{\sigma(i)} = B'_i$ for $1 \leq i \leq k$. Hence any $M \in \mathcal{A} \circ \mathcal{B}$ possesses a core structure A and *components* $B_1, \dots, B_k$. The size is then given by $|M| := \sum_{1 \leq i \leq k} |B_i|$. The respective generating series fulfils

$$(\mathcal{A} \circ \mathcal{B})(x) = \mathcal{A}(\mathcal{B}(x)).$$

We also write $\mathcal{A} \circ \mathcal{B} = \mathcal{A}(\mathcal{B})$.

A.2 $\mathcal{R}$-enriched trees

The theory of this subsection is extensively treated in [5, Chapter 3]. Enriched trees are rooted labelled unordered trees, where the offspring set of each vertex is decorated with an additional structure. More precisely, let $\mathcal{A}$ be the class of rooted labelled unordered trees (the Cayley trees from Example A.1 where one distinct node is chosen as a root) and $\mathcal{R}$ be an arbitrary combinatorial class. The class of $\mathcal{R}$-enriched trees $\mathcal{A}_{\mathcal{R}}$ contains all sequences

$$(T, (R_v)_{v \in T}), \qquad T \in \mathcal{T}, R_v \in \mathcal{R}[P_v], v \in T,$$

where P_v is the label set of the offspring of vertex $v \in T$ and the sequence $(R_v)_{v \in T}$ is canonically ordered, for example in breath first search appearance of $v \in T$. As any $\mathcal{R}$-enriched tree is comprised of a root to which an $\mathcal{R}$-structure is attached whose atoms are replaced by $\mathcal{R}$-enriched trees, we obtain the functional equation

$$\mathcal{A}_{\mathcal{R}} = \mathcal{X} \cdot \mathcal{R}(\mathcal{A}_{\mathcal{R}}).$$

This is a combination of the product and substitution construction and so we get for the generating series

$$\mathcal{A}_{\mathcal{R}}(x) = x\mathcal{R}(\mathcal{A}_{\mathcal{R}}(x)). \tag{A.1}$$

Note that $\mathcal{A}_{\mathcal{R}}$ consists at least of the root in $\mathcal{X}$ so that the substitution construction is well-defined.

Example A.2. *Letting $\mathcal{R}$ be one of the basic combinatorial classes, we retrieve basic models of trees. Note that we consider the labelled setting which may feel a bit unnatural when considering trees embedded in the plane since nodes are on the one hand identified by their ordering and in addition by the labels they bear. For example, rooted ordered trees possess no non-trivial automorphisms which is why every unlabelled tree of size n leads to $n!$ labelled trees. But for the sake of understanding the construction of enriched trees, we deem the following simple examples to be helpful.*

- *By choosing $\mathcal{R} = \mathrm{SET}$ no additional structure is imposed on the offspring set of each vertex, so that we obtain that $\mathcal{A}_{\mathrm{SET}} = \mathcal{A}$.*
- *When $\mathcal{R} = \mathrm{SEQ}$ the offspring of each vertex is given an ordering and we obtain that $\mathcal{A}_{\mathrm{SEQ}} = \mathcal{T}$, the class of rooted labelled ordered trees.*
- *By cyclically ordering the offspring of each vertex (that is considering $\mathcal{R} = \mathrm{CYC}$) we obtain the class $\mathcal{A}_{\mathrm{CYC}} = \mathcal{P}$ of rooted labelled plane trees.*

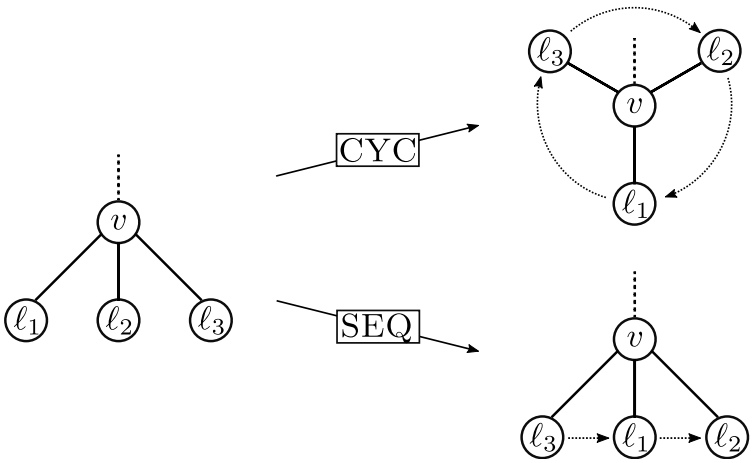

Figure 2: An $\mathcal{R}$-enriched tree observed locally at some node v and its offspring labelled by $\{\ell_1, \ell_2, \ell_3\}$ paired with the $\mathcal{R}$-object R_v, where $R_v = (\ell_3, \ell_2, \ell_1) \in \mathsf{SEQ}[\{\ell_1, \ell_2, \ell_3\}]$ or $(\ell_2, \ell_1, \ell_3)_{\simeq} \in \mathsf{CYC}[\{\ell_1, \ell_2, \ell_3\}]$.

A.3 Boltzmann Random variables

The Boltzmann framework for the labelled framework was introduced in [19]. Given a class $\mathcal{C}$ and a parameter $y > 0$ such that $\mathcal{C}(y) < \infty$ we define the corresponding Boltzmann random variable $\Gamma\mathcal{C}(y)$ by

$$\mathbb{P}(\Gamma\mathcal{C}(y) = C) = \frac{y^{|C|}}{\mathcal{C}(y)|C|!}, \qquad C \in \mathcal{C}.$$

The random variable $\Gamma\mathcal{C}(y)$ is taking values in the entire space $\mathcal{C}$. Set $\mathsf{C}_n := (\Gamma\mathcal{C}(y) \mid |\Gamma\mathcal{C}(y)| = n)$. Since objects of the same size have the same probability we obtain that C_n induces the uniform distribution on $\mathcal{C}_n$, i.e.

$$\mathbb{P}(\mathsf{C}_n = C) = \frac{1}{|\mathcal{C}_n|}, \qquad C \in \mathcal{C}_n, n \in \mathbb{N}_0.$$

B Proof of Lemma 3.1

In the following we first recall results concerning simply generated trees discussed thoroughly in [25] and then proceed to the proof of Theorem 3.1. Denote the class of rooted unlabelled ordered trees by $\widetilde{\mathcal{T}}$. Note that $\widetilde{\mathcal{T}}$ is obtained by taking equivalence classes under relabelling in $\mathcal{T}$, the labelled version discussed in Example A.2. Let $\omega = (\omega_k)_{k\geq 0}$ be a real-valued non-negative sequence and for any given finite ordered tree T in $\widetilde{\mathcal{T}}$ define its weight by

$$\omega(T) := \prod_{v\in T} \omega_{d^+(v)}.$$

In order to receive non-trivial objects it is useful to assume $\omega_0 > 0$ and $\omega_k > 0$ for some $k \geq 2$. Define by T_n the n-sized random tree with distribution

$$\mathbb{P}(\mathsf{T}_n = T) = \frac{\omega(T)}{Z_n}, \qquad \text{where } T \in \widetilde{\mathcal{T}}_n, Z_n = \sum_{T'\in\widetilde{\mathcal{T}}_n} \omega(T'). \tag{B.1}$$

We only consider values for n such that the so-called partition function Z_n is strictly greater than 0. If ω is a probability weight sequence we notice that T_n is the size-constrained Galton-Watson tree with this distribution. Thus simply generated trees are a generalization of Galton-Watson trees. However, due to the technique of *tilting* sequences, it is often possible to view T_n as a Galton-Watson tree if ω is not a probability sequence. In what follows we clarify in which cases it is possible to transform ω to a probability sequence such that the distribution of T_n is not altered and in addition the underlying offspring distribution is critical, meaning that the mean is 1, and of finite variance.

Denote the radius of convergence of $\Phi(x) := \sum_{k\geq 0} \omega_k x^k$ by ρ. Further set

$$\Psi(x) := \frac{x\Phi'(x)}{\Phi(x)} \qquad \text{and} \qquad \nu := \lim_{x\to\rho} \Psi(x) \in (0, \infty].$$

The generating function $Z(x) = \sum_{n\geq 1} Z_n x^n$ of the partition function is recursively given by $Z(x) = x\Phi(Z(x))$ and has radius of convergence

$$\rho_Z = \frac{\tau}{\Phi(\tau)}, \qquad \text{where } \tau = Z(\rho_Z). \tag{B.2}$$

When $\nu > 1$ or $0 < \tau < \rho$ the modified sequence

$$\pi_k := \frac{\omega_k \tau^k}{\Phi(\tau)} \tag{B.3}$$

is a probability sequence with mean 1 and finite exponential moments. The same modification can be done for $0 < \nu \leq 1$ with the result that $(\pi_k)_{k\geq 0}$ has either mean 1 but infinite variance ($\nu = 1$) or mean strictly smaller than 1 ($\nu < 1$). If $\nu = 0$ it is not possible at all to define a probability sequence equivalent to ω. In this work we stick to the condition $\nu > 1$ or $0 < \tau < \rho$ which corresponds to the case (Ia) in [25]. With this at hand, we are able to prove Lemma 3.1.

Proof of Lemma 3.1. Let $\mathsf{T}_n(\omega)$ be the simply generated tree of size n with weight sequence $\omega = (r_k/k!)_{k\geq 0}$ defined in (B.1), i.e.

$$\mathbb{P}(\mathsf{T}_n(\omega) = T) = \frac{\omega(T)}{Z_n}, \qquad \text{where } \omega(T) = \prod_{v\in T} \frac{r_{d^+(v)}}{d^+(v)!}, Z_n = \sum_{T'\in\widetilde{\mathcal{T}}_n} \omega(T'), T \in \widetilde{\mathcal{T}}_n. \tag{B.4}$$

Then the generating function $Z(x)$ of the partition functions $(Z_n)_{n\geq 0}$ is recursively given by $Z(x) = x\mathcal{R}(Z(x))$. Let $(\mathcal{A}_\mathcal{R})_n$ be the collection of objects in $\mathcal{A}_\mathcal{R}$ of size n. Since $\mathcal{A}_\mathcal{R} = \mathcal{X}\cdot\mathcal{R}(\mathcal{A}_\mathcal{R})$ we know by (A.1) that $\mathcal{A}_\mathcal{R}(x) = x\mathcal{R}(\mathcal{A}_\mathcal{R}(x))$ by which we deduce

$$\mathcal{A}_\mathcal{R}(x) = Z(x) \qquad \text{and} \qquad \frac{|(\mathcal{A}_\mathcal{R})_n|}{n!} = Z_n, \quad n \in \mathbb{N}_0. \tag{B.5}$$

We conclude that due to (B.2) the radius of convergence $\rho_{\mathcal{A}_\mathcal{R}}$ of $\mathcal{A}_\mathcal{R}(x)$ (or equivalently $Z(x)$) is given by

$$\rho_{\mathcal{A}_\mathcal{R}} = \frac{\tau}{\mathcal{R}(\tau)}, \qquad \text{where } \tau = \mathcal{A}_\mathcal{R}(\rho_{\mathcal{A}_\mathcal{R}}).$$

Assumption (A) immediately implies that $0 < \tau < \rho_\mathcal{R}$ and we deduce that $(\pi_k)_{k\geq 0}$ as defined in (B.3) is the probability distribution with mean 1 and finite exponential moments such that the distribution of $\mathsf{T}_n(\omega)$ is not altered by switching to $(\pi_k)_{k\in\mathbb{N}_0}$. Further we observe that this is also the distribution of ξ given in (1.1), i.e.

$$\pi_k = \frac{r_k\tau^k}{\mathcal{R}(\tau)k!} = \frac{r_k\mathcal{A}(\rho_\mathcal{A})^k}{\mathcal{R}(\mathcal{A}(\rho_\mathcal{A}))k!} = p_k, \quad k \in \mathbb{N}_0$$

and hence the claim is verified. □

C Galton–Watson trees conditioned on the number of nodes whose outdegree lies in a given set

Throughout this section we fix a proper subset $\Omega \subsetneq \mathbb{N}_0$ satisfying $0 \in \Omega$. See Remark C.3 below for comments on this assumption. We let $\Omega^c := \mathbb{N}_0 \setminus \Omega$ denote its complement. Let ζ be a random non-negative integer satisfying

$$\mathbb{P}(\zeta = 0) > 0, \quad \mathbb{P}(\zeta \geq 2) > 0, \quad \text{and} \quad \mathbb{P}(\zeta \in \Omega) > 0. \tag{C.1}$$

We would like to generate a tree A_n^Ω that is distributed like a ζ-Galton–Watson A tree conditioned on having n vertices with outdegree in Ω. That is, letting $L_\Omega(\cdot)$ denote the number of vertices with outdegree in Ω, we set $\mathsf{A}_n^\Omega = (\mathsf{A} \mid L_\Omega(\mathsf{A}) = n)$. Of course, we only consider integers n for which this is well defined, that is, where $\mathbb{P}(L_\Omega(\mathsf{A}) = n) > 0$. Furthermore, in order to obtain a procedure that samples in linear time, we assume that

$$\mathbb{E}[\zeta] = 1 \qquad \text{and} \qquad \mathbb{E}[(1+\epsilon)^\zeta] < \infty \quad \text{for some } \epsilon > 0. \tag{C.2}$$

We also assume that the weight $\mathbb{P}(\zeta = k)$ may be computed in time $\exp(o(k))$ for each $k \geq 0$, and that the probability $\mathbb{P}(\zeta \in \Omega)$ is also given.

We could take a naive approach and sample independent copies of A until the event $L_\Omega(\cdot) = n$ takes place. However, it is known (see Kortchemski [28, Thm. 8.1]) that

$$\mathbb{P}(L_\Omega(\mathsf{A}) = n) \sim cn^{-3/2} \tag{C.3}$$

for some constant $c > 0$. Hence the expected number of copies we would have to sample is $\Theta(n^{3/2})$. Moreover, the expected time for generating a single independent copy of A and determining whether it satisfies $L_\Omega(\cdot) = n$ is $\Theta(\sqrt{n})$, assuming that we may sample an independent copy of ζ in constant time. Thus, with the naive approach we arrive at a $\Theta(n^2)$ complexity for generating A_n^Ω. This performance is not optimal, hence our motivation for describing a generator that accomplishes this in expected linear time.

The procedure we are going to describe is based on the fact that rooted trees satisfying $L_\Omega(\cdot) = n$ correspond bijectively to n-vertex $\mathcal{R}$-enriched trees for a specific class $\mathcal{R}$, see [20, 34]. Since we may generate $\mathcal{R}$-enriched trees in expected linear time via Algorithm 2.1, and since the transformation to a plane tree with $L_\Omega(\cdot) = n$ also takes expected linear time, we will arrive at generator for A_n^Ω that runs in expected time $O(n)$. To be fully precise, we will use a straight-forward extension of Algorithm 2.1 to weighted species, because A_n^Ω is not (necessarily) uniform among all plane trees A with $L_\Omega(A) = n$. Instead, it assumes such a tree A with probability proportional to a certain multiplicative weight given by

$$\mathbb{P}(\mathsf{A}_n^\Omega = A) = \frac{\mathbb{P}(\mathsf{A} = A)}{\mathbb{P}(L_\Omega(\mathsf{A}) = n)} = \frac{1}{\mathbb{P}(L_\Omega(\mathsf{A}) = n)} \prod_{v \in A} \mathbb{P}(\zeta = d_A^+(v)). \tag{C.4}$$

Consequently, the random n-vertex $\mathcal{R}$-enriched tree corresponding to A_n^Ω is not (necessarily) uniform. Instead, it assumes an enriched tree corresponding to A with the same probability that A_n^Ω assumes A. Furthermore, again to be fully precise, Algorithm 2.1 is formulated for labelled structures. The plane trees we generate here are asymmetric unlabelled structures. That is, it is irrelevant whether we consider them as labelled or unlabelled, since they have no non-trivial symmetries. Thus, we may safely ignore labels in this section. Algorithm 2.1 still applies.

Let us start with the description of the weighted species $\mathcal{R}$ in question. For each integer $k \geq 0$ we let $\mathcal{R}_k$ denote the collection of all tuples $R = (y, x_1, \dots, x_\ell)$ satisfying $\ell \geq 0$, $y \in \Omega$, $x_1, \dots, x_\ell \in \Omega^c - 1$, and $y + \sum_{i=1}^{\ell} x_i = k$. To each such tuple R we assign a weight $\gamma(R)$ by

$$\gamma(R) = \mathbb{P}(\zeta = y) \prod_{i=1}^{\ell} \mathbb{P}(\zeta = x_i + 1). \tag{C.5}$$

It was shown in [20, 34] in a more general context that an ordered rooted tree A with $L_\Omega(A) = n$ corresponds bijectively to a pair (T, β) of an ordered rooted tree T with n vertices, and a map β that assigns to each inner vertex $v \in T$ a structure $\beta(v) \in \mathcal{R}_{d_T^+(v)}$. Here A is constructed from T by a blow-up procedure that replaces a vertex $v \in T$ and the edges to its children by a tree constructed from $\beta(v)$ as illustrated in Figure 3.

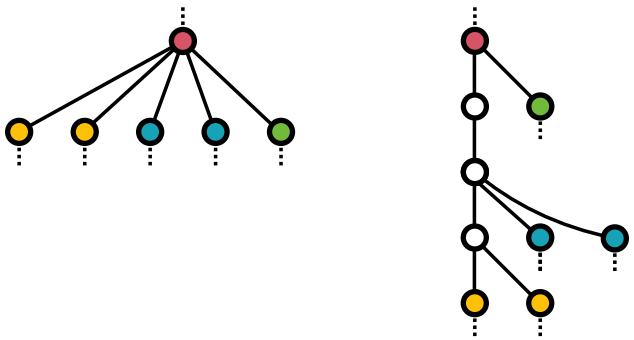

Figure 3: Blow-up procedure of a vertex v (red) having 5 children and decoration $\beta(v) = (2, 2, 0, 1)$.

This correspondence is weight-preserving in the sense that the weight $\mathbb{P}(\mathsf{A} = A)$ of the tree A equals the weight $\prod_{v \in T} \gamma(\beta(v))$ of the decorated tree (T, β). Furthermore, the random decorated tree (T_n, β_n) corresponding to the random tree A_n^Ω has the property, that T_n is distributed like a ξ-Galton–Watson tree conditioned on having n vertices, with the random integer $\xi \geq 0$ given by

its probability generating function

$$\mathbb{E}[z^{\xi}] = \left(\sum_{k\in\Omega} \mathbb{P}(\zeta = k) z^k\right) \left(1 - \sum_{k\in\Omega^c} \mathbb{P}(\zeta = k) z^{k-1}\right)^{-1}. \tag{C.6}$$

Conditional on T_n, each decoration $\beta_n(v)$ gets drawn from $\mathcal{R}_{d^+_{\mathsf{T}_n}(v)}$ with probability proportional to its γ-weight, independently from the rest. See [34, 38] for detailed justifications.

Our strategy for generating A^{Ω}_n in expected time $O(n)$ is to generate the $\mathcal{R}$-enriched plane tree (T_n, β_n) according to Algorithm 2.1 and apply the blow-up procedure. In order to verify that this works we have to do two things. First, we have to check that the weighted analogues of the conditions of Algorithm 2.1 are met. Second, we have to check that the expected time for applying the blow-up procedure to (T_n, β_n) is $O(n)$. Note that there is a subtle difficulty in the second step, because if $\mathbb{P}(\zeta = 1) > 0$ the number of vertices of A^{Ω}_n may be arbitrarily large and the blow-up procedure may take arbitrarily long. Hence we really have to bound the expected duration.

Let us start by verifying the conditions of Algorithm 2.1. Setting $|\mathcal{R}_k|_\gamma = \sum_{R\in\mathcal{R}_k} \gamma(R)$ for each $k \geq 0$, the ordinary generating series of the species $\mathcal{R}$ is given by

$$\mathcal{R}(z) := \sum_{k\geq 0} |\mathcal{R}_k|_\gamma z^k. \tag{C.7}$$

Condition (C.2) entails that $\mathcal{R}(z)$ has radius of convergence $\rho_{\mathcal{R}} > 1$. For any parameter $t > 0$ with $\mathcal{R}(t) < \infty$ we define a Boltzmann sampler $\Gamma\mathcal{R}(t)$ with distribution

$$\mathbb{P}(\Gamma\mathcal{R}(t) = R) = \frac{\gamma(R) t^{k(R)}}{\mathcal{R}(t)}, \qquad R \in \bigsqcup_{k\geq 0} \mathcal{R}_k \tag{C.8}$$

for $k(R) \geq 0$ the unique integer with $R \in \mathcal{R}_{k(R)}$.

Algorithm C.1. *A Boltzmann sampler $\Gamma\mathcal{R}(t)$:*

1. *Generate a random integer y with probability generating function $\mathbb{E}[(zt)^{\zeta}, \zeta \in \Omega]/\mathbb{E}[t^{\zeta}, \zeta \in \Omega]$.*
2. *Generate a random integer ℓ following a geometric distribution with parameter $\mathbb{P}(\zeta \in \Omega^c)$. That is, its probability generating function equals $(1 - \mathbb{P}(\zeta \in \Omega^c))/(1 - z\mathbb{P}(\zeta \in \Omega^c))$.*
3. *For each $1 \leq i \leq \ell$ generate a random integer $x_i \geq 0$ with probability generating function given by $\mathbb{P}(\zeta \in \Omega^c)^{-1} \sum_{k\in\Omega^c} \mathbb{P}(\zeta = k)(zt)^{k-1}$.*
4. *Return $(y, x_1, \ldots, x_\ell)$.*

For any integer $k \geq 0$ (satisfying $|\mathcal{R}_k|_\gamma > 0$) we may condition $\Gamma\mathcal{R}(t)$ on returning an element from $\mathcal{R}_k$. The element generated in this way is drawn with probability proportional to its γ-weight from $\mathcal{R}_k$. Since all coordinates of a tuple from $\mathcal{R}_k$ are at most k, we may fix some $K \geq k$ and work with a truncated version $\Gamma_{\leq K}\mathcal{R}(t)$ instead. That is, $\Gamma_{\leq K}\mathcal{R}(t)$ uses truncated versions $(y \mid y \leq K)$ and $(x_i \mid x_i \leq K)$ instead, and conditioning $\Gamma_{\leq K}\mathcal{R}(t)$ on producing an element from $\mathcal{R}_k$ also yields a random element that gets drawn with probability proportional to its γ-weight. Furthermore, constructing $\Gamma_{\leq K}\mathcal{R}(t)$ only requires knowledge of $\mathbb{P}(\zeta \in \Omega)$ and the probabilities $\mathbb{P}(\zeta = i)$ for $0 \leq i \leq K$. We assumed that $\mathbb{P}(\zeta \in \Omega)$ is given and that $\mathbb{P}(\zeta = i)$ may be computed in $e^{o(i)}$ steps. Hence constructing $\Gamma_{\leq K}\mathcal{R}(t)$ requires $e^{o(K)}$ preprocessing time. Running a single instance of $\Gamma_{\leq K}\mathcal{R}(t)$ only requires constant time. Moreover, $\mathbb{P}(\Gamma_{\leq K}\mathcal{R}(t) \in \mathcal{R}_k) \geq \mathbb{P}(\Gamma\mathcal{R}(t) \in \mathcal{R}_k)$, hence generating a random element from $\mathcal{R}_k$ using $\Gamma_{\leq K}\mathcal{R}(t)$ is at least as fast as using $\Gamma\mathcal{R}(t)$.

We may now state our final algorithm for sampling A^{Ω}_n.

Algorithm C.2. *A generator for A_n^Ω that runs in expected time $O(n)$.*

1. *Use Algorithm 2.3 to sample a Galton-Watson tree T_n with offspring distribution ξ conditioned on having n vertices.*
2. *Let $K := \Delta(\mathsf{T}_n)$ denote the maximal outdegree of T_n. For a fixed $1 < t_0 < \rho_{\mathcal{R}}$ repeatedly call for each $v \in \mathsf{T}_n$ the sampler $\Gamma_{\leq K}\mathcal{R}(t_0)$ until it produces an object $\beta_n(v)$ from $\mathcal{R}_{d^+_{\mathsf{T}_n}(v)}$.*
3. *Perform the blow-up procedure illustrated in Figure 3 on (T_n, β_n) to produce the tree A_n^Ω.*

Here's a justification why Algorithm C.2 runs in expected time $O(n)$:

Proof. The expression of $\mathbb{E}[z^\xi]$ in Equation (C.6) allows us to compute $\mathbb{P}(\xi = k)$ in $e^{o(k)}$ steps for any $k \geq 0$, since we assumed $\mathbb{P}(\zeta = k)$ to be computable in in $e^{o(k)}$ steps. Furthermore, using $\mathbb{E}[\zeta] = 1$ it follows from Equation (C.6) that $\mathbb{E}[\xi] = 1$, see [34, Thm. 6] for details on the calculation. Moreover, ξ has finite exponential moments. Thus, Algorithm 2.3 produces the tree T_n in expected time $O(n)$. Calculating the maximum degree $K := \Delta(\mathsf{T}_n)$ also takes $O(n)$ steps. The unique generating series $\mathcal{A}_{\mathcal{R}}(z)$ with $\mathcal{A}_{\mathcal{R}}(z) = z\mathcal{R}(\mathcal{A}_{\mathcal{R}}(z))$ satisfies $[z^n]\mathcal{A}_{\mathcal{R}}(z) = \mathbb{P}(L_\Omega(\mathsf{A}) = n)$. By Equation (C.3) it follows that it has radius of convergence $\rho_{\mathcal{A}_{\mathcal{R}}} = 1$ and hence $\mathcal{A}_{\mathcal{R}}(\rho_{\mathcal{A}_{\mathcal{R}}}) = 1$. We observed above that $\rho_{\mathcal{R}} > 1$. Thus, as justified in the proof of Thm. 1.5, we may generate the decoration β_n in expected time $O(n)$ by repeatedly running $\Gamma_{\leq K}\mathcal{R}(t)$ for each vertex $v \in \mathsf{T}_n$ until we generate an element from $\mathcal{R}_{d^+_{\mathsf{T}_n}(v)}$. As for the blow-up procedure, the time required for performing the blow-up of a vertex $v \in \mathsf{T}_n$ with decoration $\beta_n(v) = (y, x_1, \ldots, x_\ell)$ is bounded by $O(d^+_{\mathsf{T}_n}(v) + \ell)$. Recall that the outdegrees of a tree with n vertices sum up to $n-1$. Summing over the n vertices of T_n, the total time for performing all blow-up operations is hence bounded by a constant multiple of n plus the sum of n independent copies of a geometric random variable. Taking the expectation, it follows that the expected time for performing the blow-up is $O(n)$. Hence the total expected time for generating A_n^Ω using Algorithm C.2 is $O(n)$. □

Remark C.3. *Throughout, we assumed that $0 \in \Omega$. Rizzolo's [34] methods may be used to generalize the procedure so that this assumption is no longer necessary. However, the decorations and the blow-up procedure are far more technical in the case $0 \notin \Omega$. We leave the details of to the reader, because all applications of the present section to models of combinatorial structures considered below are already covered by the special case $\Omega = \{0\}$.*