Entropy and Optimal Compression of Some General Plane Trees

ZBIGNIEW GOŁĘBIEWSKI, Wrocław University of Science and Technology, Poland

ABRAM MAGNER, Center for the Science of Information, USA

WOJCIECH SZPANKOWSKI, Purdue University, USA

We continue developing the information theory of structured data. In this article, we study models generating d -ary trees ($d \geq 2$) and trees with unrestricted degree. We first compute the entropy which gives us the fundamental lower bound on compression of such trees. Then we present efficient compression algorithms based on arithmetic encoding that achieve the entropy within a constant number of bits. A naïve implementation of these algorithms has a prohibitive time complexity of $O(n^d)$ elementary arithmetic operations (each corresponding to a number $f(n, d)$ of bit operations), but our efficient algorithms run in $O(n^2)$ of these operations, where n is the number of nodes. It turns out that extending source coding (i.e., compression) from sequences to advanced data structures such as degree-unconstrained trees is mathematically quite challenging and leads to recurrences that find ample applications in the information theory of general structures (e.g., to analyze the information content of degree-unconstrained non-plane trees).

CCS Concepts: • **Mathematics of computing** → **Trees; Information theory**; • **Theory of computation** → **Data compression**; *Design and analysis of algorithms*;

Additional Key Words and Phrases: Arithmetic coding, random trees, plane recursive trees

ACM Reference format:

Zbigniew Gołębiewski, Abram Magner, and Wojciech Szpankowski. 2018. Entropy and Optimal Compression of Some General Plane Trees. *ACM Trans. Algorithms* 15, 1, Article 3 (October 2018), 23 pages.

<https://doi.org/10.1145/3275444>

1 INTRODUCTION

Advances in sensing, communication, and storage technologies have created a state of the art in which our ability to collect data from richly instrumented environments has far outpaced our ability to process, understand, and analyze this data in a (provably) rigorous manner. A significant component of this complexity arises from the highly structured nature of data. This poses significant challenges for theoretical characterization of limits of information storage, content, and

This work was supported in part by NSF Center for Science of Information (CSol) Grant CCF-0939370, by NSF Grant CCF-1524312, NIH Grant 1U01CA198941-01, and by Polish NCN Grant 2013/09/B/ST6/02258.

Authors' addresses: Z. Gołębiewski, Wrocław University of Science and Technology, Faculty of Fundamental Problems of Technology, Wrocław, Poland; email: zbigniew.golebiewski@pwr.edu.pl; A. Magner, Center for the Science of Information, Purdue University, West Lafayette, IN; email: abram10@gmail.com; W. Szpankowski, Purdue University, Computer Science, West Lafayette, IN; email: spa@cs.purdue.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Association for Computing Machinery.

1549-6325/2018/10-ART3 \$15.00

<https://doi.org/10.1145/3275444>

transmission and methods that achieve these limits. While heuristic approaches are often currently used, critical issues regarding their performance remain. These challenges have motivated our recent research program [5, 6, 21] and others [1, 18, 26], which aims to characterize limits on information content of non-sequential data, such as trees and graphs. Quite generally, these problems consist of two complementary parts: fundamental achievable limits on data compression (where the Shannon entropy of the probability distribution from which the data arises gives the minimum possible expected code length), and, conversely, efficient encoding/decoding algorithms that achieve those limits.

Briefly, the entropy of a discrete random variable X taking values on a domain $\mathcal{X}$ is, properly speaking, a functional of the *distribution* of X . It is denoted by $H(X)$ and is given by

$$H(X) = - \sum_{x \in \mathcal{X}} \mathbb{P}(X = x) \log(\mathbb{P}(X = x)).$$

The entropy of a random variable is of fundamental importance for several reasons, among which is its connection to compression: a *source code* (or compression code) is a pair of mappings $(e : \mathcal{X} \rightarrow S \subseteq \{0, 1\}^*, d : S \rightarrow \mathcal{X})$, such that for all $x \in \mathcal{X}$, $d(e(x)) = x$, and for every $x, y \in \mathcal{X}$, $e(x)$ is not a prefix of $e(y)$. The first mapping $e(x)$ gives a bit string, which is the compressed representation of x , and the mapping $d(\cdot)$ decompresses it. The entropy gives the minimum achievable expected code length (i.e., $\mathbb{E}[|e(x)|]$) for a source code for X . In our case, we are interested in entropy and efficient compression for distributions on sets $\mathcal{X}$ of combinatorially structured objects, such as trees and graphs. It is a classical abuse of notation to denote, for a random variable T , the entropy of its distribution by $H(T)$ (despite the entropy *not* being a function of the value of a random variable). Throughout this article, we will follow this convention. By the entropy of a certain type of tree, we actually mean the entropy of a specified distribution on trees, and not any sort of topological index defined on an individual tree.

As a start to understanding structured data in an information-theoretic setting, we focused on graphs [5] and trees with vertex (correlated) names [21]. In 1990, Naor proposed an efficiently computable representation for unlabeled graphs (solving Turán's open question) that is optimal up to the first two leading terms of the entropy when all unlabeled graphs are equally likely. Naor's result is asymptotically a special case of recent work of Choi and Szpankowski [5], who extended Naor's result to general Erdős-Rényi graphs. In particular, in [5] the entropy and an optimal compression algorithm (up to two leading terms of the entropy) for Erdős-Rényi graph structures were presented. Recently, these results were extended to preferential attachment graphs in [20] and to uniform random intersection graphs in [15]. Furthermore, in [22] an automata approach was used to design an optimal graph compression scheme. For unlabeled binary plane increasing trees rigorous information-theoretic results were obtained in [18], complemented by a universal grammar-based lossless coding scheme [26]. Finally, in recent work [21] (see also [6]) the authors study binary trees with structure-correlated vertex names, as well as non-plane binary increasing trees, and design optimal compression schemes based on arithmetic encoding.

In this article, we study three different models generalizing unlabeled binary plane increasing trees: d -ary plane increasing trees without correlated vertex names (i.e., trees with degree $d \geq 2$), m -ary search trees, and degree-unconstrained trees without any restriction on vertex degrees, giving both precise entropy calculations and information theoretically optimal and efficient compression algorithms. In particular, we study their unlabeled versions. It is well known (see [2]) that for binary trees an equivalence was proved between two models: the *binary search tree model* and a model in which leaves are selected randomly to expand the tree by adding two additional nodes (new leaves). This equivalence allowed one to analyze the entropy of such trees by solving a relatively simple recurrence, and then to construct an optimal compression algorithm. Unfortunately,

this equivalence does not work for $d \geq 3$, and we develop here a different methodology to overcome this problem.

The tree models that we consider here have a long history, and various aspects of their asymptotic properties have been studied using a variety of techniques (for instance, generating functions and complex analytic techniques, Pólya urn models). See, for example, [3, 8, 9, 13, 17, 23] for some representative works. One of the most relevant ones to our work is [23], which studies the asymptotic distribution and moments of additive functionals on d -ary plane increasing trees. The results in that work apply to our case, since the entropy of a random tree may be written as the expected value of an additive functional—namely, $\log p(T)$, where $p(T)$ denotes the probability assigned to the tree in question under our model. Thus, our exact and asymptotic entropy results (but not our optimal compression results, which are novel) for d -ary plane increasing trees are direct consequences of their framework (similarly, the asymptotics in [25] apply for plane increasing trees, and those in [4, 12] apply for m -ary search trees). We include alternative, elementary derivations for the sake of completeness.

Another relevant work is [13], which studies the asymptotic number of occurrences of given subtree patterns in binary plane increasing trees. They include results on the size of a directed, acyclic graph representation of such trees (which captures the repeated occurrence of subtree patterns in a given tree). This is intimately related to more recent works on universal compression of binary trees [26].

In the present work, we shall show that for the models (such as the random unlabeled d -ary plane increasing tree¹ and m -ary search tree models described in the present article) under consideration that produce d -ary trees T_n on n internal nodes the entropy $H(T_n)$ satisfies

$$H(T_n) = H(\text{root}) + d \sum_{k=0}^{n-1} H(T_k) p_{n,k},$$

where $H(\text{root})$ is the entropy of the split probability distribution at the root (i.e., the distribution of the d -tuple of root subtree sizes), and $p_{n,k}$ is the probability of one specified subtree being of size k (a similar recurrence holds for unlabeled random plane increasing trees with no degree restrictions: the second term in the above recurrence becomes a sum over all possible root degrees). The distribution of the split at the root and the values for $p_{n,k}$ differ depending on the model under consideration. For the m -ary search tree model discussed in Section 2, this recurrence can be handled by results from [4, 12]. Then we consider the more interesting (and possibly more relevant in the context of compression) d -ary plane increasing tree model in which we randomly select a leaf and add exactly d leaves to it as children. In [21] $d = 2$ was studied, but the analysis is somewhat more complicated when $d > 2$ (though both cases can be treated uniformly when only asymptotics are desired). For example, for $d = 2$ we have $p_{n,k} = 1/n$ while for $d = 3$ we can prove that

$$p_{n,k} = \frac{1}{2n} \frac{\binom{2k}{k} 2^{2n}}{\binom{2n}{n} 2^{2k}}.$$

Note the dependence on k . We complete our analysis by studying degree-unconstrained trees in which there is no restriction on the node degree.

¹An unlabeled increasing tree is at first glance an oxymoron: we mean by this a random increasing tree whose labels have been removed (i.e., we actually consider the equivalence class of the random labeled tree, where two such trees are equivalent if one can be gotten from the other by a root-preserving permutation).

In Section 3, we show that for unlabeled d -ary plane increasing trees we need to analyze a recurrence of the following form (see Lemma 3.4):

$$x_n = a_n + \frac{\alpha}{n} \frac{n!}{\Gamma(n + \alpha - 1)} \sum_{k=0}^{n-1} \frac{\Gamma(k + \alpha - 1)}{k!} x_k, \quad (1)$$

where $\alpha = d/(d - 1)$, a_n is a given sequence, and Γ is the Euler gamma function. This is a recurrence with full history and is amenable to what is called the *method of differences*, by which such a recurrence can be converted to a linear one of the form $x_n = A_n x_{n-1} + B_n$, for some sequences A_n , B_n . In the case of $d \geq 3$, this step is rather complicated and, furthermore, the specific form of A_n and B_n leads to a “closed-form” solution which makes subsequent asymptotic analysis nontrivial (this is reflected in the forms of the singularities of the respective generating functions of $\{x_n\}$ for different values of d : when $d = 2$, the generating function is meromorphic, while for $d \geq 3$, algebraic singularities are introduced). The situation is even more involved when we consider degree-unconstrained trees in Section 3.4 where no restrictions on degrees are imposed.

After describing in Section 2 the three generalizations of the binary tree model from [21], we present our main results in Section 3. We first provide in Corollary 3.2 the entropy rate for m -ary search trees. Then we consider d -ary plane increasing trees and in Theorem 3.5 give our expression for the entropy of such trees. We extend it to degree-unconstrained plane increasing trees in Theorem 3.9. After establishing these fundamental lower bounds on tree compression, we provide efficient compression algorithms, which are optimal in expected code length to within a constant number of bits. The algorithms are a variation of the *arithmetic coding* technique [7], wherein a tree T to be compressed is mapped to a subinterval of $[0, 1]$ (in such a way as to avoid any overlap with the subintervals corresponding to other trees of the same size), whose length is equal to the probability of T . The binary codeword for T is then a truncation of the binary expansion of the midpoint of the interval of T . A naive implementation of this idea yields an algorithm whose time complexity scales like $\Theta(n^d f(d, n))$ in the worst case for d -ary trees, for some positive function f . However, by careful rearrangement of expressions involved in interval calculations, we are able to reduce this complexity to $\Theta(n^2 f(d, n))$, a substantial improvement. We furthermore give experimental results which show that the typical compressed code lengths for our codes are very close to their expected values—within four bits for trees with 500 internal nodes.

We note that the present article is an expanded journal version of [16].

2 MODELS

In this section, we describe the concepts of unlabeled plane trees with and without restrictions on the nodes’ out-degrees. This will allow us to introduce three models of tree generation.

We call a rooted tree a plane tree when we distinguish left-to-right order of the successors of each node; i.e., we distinguish all different embeddings of a tree into the plane (see [10]). Degree-unconstrained unlabeled plane trees are rooted plane trees with no restriction on the number of children of the nodes. On the other hand, unlabeled d -ary plane trees are rooted plane trees where each internal (i.e., non-leaf) node has exactly d children (either internal or external). Since they are unlabeled, they can be seen as objects that encode the structures of a possible labeled (in the graph-theoretic sense) plane tree.

2.1 Unlabeled m -Ary Search Tree Model

Search trees are plane trees built from a set of n distinct keys taken from some totally ordered set, for instance, a random permutation of the numbers $\{1, 2, \dots, n\}$. An m -ary *search tree* is a tree in which each node has at most m children; moreover, each node stores up to $m - 1$ keys. We define

Fig. 1. Example of a 3-ary search tree of size 9 built from the permutation (4, 7, 3, 5, 1, 2, 9, 6, 8) and its unlabeled counterpart.

the size of a search tree as the number of keys n . The construction of m -ary search trees can be described as follows [10].

If $n = 0$, the tree is empty. If $1 \leq n \leq m - 1$, the tree consists of a root only, with all keys stored in the root. If $n \geq m$, we select $m - 1$ keys that are called pivots. The pivots are stored in the root. The $m - 1$ pivots split the set of remaining $n - m + 1$ keys into m sublists $I_1, \dots, I_m$: if the pivots are $p_1 < p_2 < \dots < p_{m-1}$, then $I_1 := (p_i : p_i < p_1)$, $I_2 := (p_i : p_1 < p_i < p_2)$, $\dots$, $I_m := (p_i : p_{m-1} < p_i)$. We then recursively construct a search tree for each of the sets I_i of keys. In order to obtain an unlabeled search tree of size n , we remove the keys from a search tree of size n (see Figure 1).

The standard probability model assumes that every permutation of the keys $\{1, \dots, n\}$ is equally likely. The choice of pivots can then be deterministic. For example, one always chooses the first $m - 1$ keys. After removing node labels from a m -ary search tree generated by a uniformly random permutation, we obtain a random unlabeled search tree, where the probability distribution of the resulting tree on the set of unlabeled search trees is non-uniform.

2.2 Unlabeled d -ary Plane Increasing Tree Model

We consider the following generation model of an unlabeled random d -ary plane increasing tree [2, 10]. The process starts with an empty tree, that is, with just an external node (leaf). The first step in the growth process is to replace this external node by an internal one with d successors that are external nodes (see Figure 2). Then with probability $\frac{1}{d}$ each, one of these d external nodes is selected and again replaced by an internal node with d successors. In each subsequent step one of the external nodes is replaced (with equal probability) by an internal node with d successors.

This evolution process (if we had labeled each node with the time index at which it arrived) produces a random d -ary plane increasing tree (see [2, 10]). By definition, d -ary plane increasing trees

Fig. 2. Example of the generation process that produces 3-ary plane tree of size 4 and its unlabeled counterpart.

are rooted plane trees with labels on internal nodes and where each node has exactly d successors (either internal or external). The root is labeled by 1 and the labels of all internal successors of any node v are larger than the label of v . We define the size of the d -ary plane increasing tree by the number of internal nodes.

Let $\mathcal{F}$ denote the set of all unlabeled d -ary plane trees and, for each integer $n \geq 0$, let $\mathcal{F}_n$ be the subset of $\mathcal{F}$ consisting of all trees that contains exactly n internal nodes. Let F_n be a random variable supported on $\mathcal{F}_n$, constructed by the generation process just described.

Similarly, let $\mathcal{G}$ and $\mathcal{G}_n$ be the analogous sets of labeled d -ary plane increasing trees. Throughout the article, we will denote a given unlabeled d -ary plane tree of size n as f_n and a given labeled d -ary plane increasing tree of size n as g_n .

Observe that the above described evolution process (choosing an external node to replace with uniform distribution among all external nodes) generates every d -ary plane increasing tree of size n in a unique way and with uniform distribution. However, after removing labels from a d -ary plane increasing tree g_n we obtain an unlabeled d -ary plane tree f_n with a non-uniform distribution. For example, there are $\binom{3}{2,0,1} \binom{1}{0,1,0} = 3$ ways to obtain the resulting tree from Figure 2, but there is only one way (exactly $\binom{3}{3,0,0} \binom{2}{2,0,0} \binom{1}{1,0,0} = 1$) to obtain a tree where every internal node is a leftmost child of a parent node.

Fig. 3. Example of the generation process that produces a labeled degree-unconstrained plane tree of size 5 and its unlabeled counterpart.

2.3 Unlabeled Degree-Unconstrained Plane Increasing Tree Model

We consider the following generation model of unlabeled plane trees. Suppose that the process starts with the root node carrying a label 1. Then we add a child node with label 2 to the root. The next step is to attach a node with label 3. However, there are three possibilities: either to add it to the root (as a left or right sibling of 2) or to the node with label 2. One proceeds further in the same way, as shown in Figure 3. At the end, we remove the labels from nodes of the tree. Observe that if a node already has out-degree k (where the descendants are ordered), then there are $k + 1$ possible ways to add a new node (this time we do not distinguish between external and internal nodes). Hence, if a plane tree already has $j - 1$ nodes, then there are precisely $2j - 3$ possibilities to attach the j th node (see Figure 3). More precisely, the probability of choosing a node of out-degree k equals $(k + 1)/(2j - 3)$. We define the size of the tree as the number of nodes.

3 MAIN RESULTS

In this section, we present our main results. In particular, we briefly address the entropy of m -ary search trees. Then we present our derivation of the recurrence for the entropy of unlabeled d -ary plane increasing trees that leads us to a formula for the entropy rate. Finally, we derive the entropy rate for degree-unconstrained trees. In the next section, we give efficient and nearly optimal compression algorithms, which achieve an expected code length within a small constant number of bits of the entropies derived here.

In all our models, the probability distribution on the relevant set of trees is non-uniform, and subtrees of the root are conditionally independent given their respective sizes. Indeed, let T_n be a random variable representing a tree t_n on n internal nodes. Assume now that at the root we split

$\mathbf{t}_n$ into d subtrees of size $k_1, \dots, k_d$, respectively, where $k_1 + \dots + k_d = n - 1$. Then the probability $\mathbb{P}(T_n = \mathbf{t}_n)$ of generating tree $\mathbf{t}_n$ in all our models satisfies

$$\mathbb{P}(T_n = \mathbf{t}_n) = \mathbb{P}(k_1, \dots, k_d) \prod_{i=1}^d \mathbb{P}(T_{k_i} = \mathbf{t}_{k_i}), \quad (2)$$

where $\mathbb{P}(k_1, \dots, k_d)$ is the probability of a split at the root of n internal nodes into subtrees of sizes $k_1, \dots, k_d$, respectively. This split probability is different for m -ary search trees, d -ary increasing trees, and degree-unconstrained trees, as we shall see in this section.

We shall use the following notation. Let $\mathbf{k}^{(d)} = (k_1, \dots, k_d)$ denote a d -dimensional vector of non-negative integers and $\|\mathbf{k}^{(d)}\| = |k_1| + \dots + |k_d|$ be its L^1 norm. Let $(k, \mathbf{k}^{(d-1)}) = (k, k_2, \dots, k_d)$ denote a d -dimensional vector with the first coordinate equal to k . We often write $\mathbf{k}$ for $\mathbf{k}^{(d)}$ when the vector dimension is obvious.

3.1 Properties of Entropy

Here we summarize the main properties of entropy that we use in this article. For more information and definitions, see [7].

The main concepts that we will need are *joint* and *conditional* entropy of pairs of random variables defined on a common probability space. The following properties will be used throughout the article:

— Trivial upper bound: For a random variable X taking values in a finite domain $\mathcal{X}$,

$$H(X) \leq \log |\mathcal{X}|, \quad (3)$$

with equality being achieved whenever X is uniformly distributed on $\mathcal{X}$.

— Chain rule: For two random variables X and Y ,

$$H(X, Y) = H(X) + H(Y|X) = H(Y) + H(X|Y). \quad (4)$$

— Independence identity: Two random variables X and Y are independent if and only if

$$H(X|Y) = H(X). \quad (5)$$

3.2 The Entropy of the Unlabeled m -ary Search Trees

Let U_n denote a random unlabeled m -ary search tree with n keys, generated according to the process described earlier. We write $\mathbf{u}_n$ for an arbitrary fixed m -ary (unlabeled) search tree with n keys.

We describe the splitting of keys at the root of the search tree by the random vector $\mathbf{Y}_n^{(m)} = (Y_{n,1}, \dots, Y_{n,m})$, where $Y_{n,j} = |I_j|$ is the number of keys that go into the j th subtree of the root. If $n \geq m - 1$, we have $Y_{n,1} + \dots + Y_{n,m} = n - m + 1$ and

$$\mathbb{P}(\mathbf{Y}_n^{(m)} = \mathbf{k}^{(m)}) = \frac{1}{\binom{n}{m-1}}. \quad (6)$$

Notice that $\mathbb{P}(\mathbf{Y}_n^{(m)} = \mathbf{k}^{(m)})$ does not depend on the vector $\mathbf{k}^{(m)}$ coordinates $k_1, \dots, k_m$, which simplifies calculations.

Suppose that the tree $\mathbf{u}_n$ has subtrees $\mathbf{u}_{k_1}, \dots, \mathbf{u}_{k_m}$ of sizes $k_1, \dots, k_m$. Then by (2),

$$\mathbb{P}(U_n = \mathbf{u}_n) = \mathbb{P}(\mathbf{Y}_n^{(m)} = \mathbf{k}^{(m)}) \prod_{j=1}^m \mathbb{P}(U_{k_j} = \mathbf{u}_{k_j}). \quad (7)$$

Let us establish the initial conditions of the entropy of m -ary search trees. If $n = 0$, we have an empty tree, and $H(U_0) = 0$. Moreover, if $1 \leq n \leq m - 1$, all keys are stored in one node, and

$H(U_n) = 0$. For $n > m - 1$, there is a bijection between a tree U_n and a tuple $(Y_n^{(m)}, U_{Y_{n,1}}, \dots, U_{Y_{n,m}})$, which is an immediate consequence of the inductive definition of m -ary search trees. Therefore, for $n > m - 1$, we have

$$\begin{aligned} H(U_n) &= H(Y_n^{(m)}, U_{Y_{n,1}}, \dots, U_{Y_{n,m}}) = H(Y_n^{(m)}) + H(U_{Y_{n,1}}, \dots, U_{Y_{n,m}} | Y_n^{(m)}) \\ &= H(Y_n^{(m)}) + m \sum_{k=0}^{n-m+1} H(U_k) \mathbb{P}(Y_{n,1} = k), \end{aligned}$$

where the second equality is by conditional independence of the root subtrees given their sizes, where two subtrees having the same size have the same distribution.

For $n \geq m - 1$ and $1 \leq j \leq m$, the random variables $Y_{n,j}$ are identically distributed, and for $0 \leq k \leq n - 1$,

$$\mathbb{P}(Y_{n,1} = k) = \sum_{\|k\|^{(m-1)} = n-m+1-k} \mathbb{P}(Y_n^{(m)} = (k, k^{(m-1)})) = \frac{\binom{n-k-1}{m-2}}{\binom{n}{m-1}}. \quad (8)$$

The last equation comes from the fact that there are $\binom{n-l-1}{m-2}$ ways of split $n - l - 1$ keys into $m - 1$ sublists (i.e., choosing $m - 2$ pivots from $n - l - 1$ keys); it also can be found, e.g., in [10].

To finish the calculation of $H(U_n)$, we need to calculate $H(Y_n^{(m)})$. From (6) we have

$$\begin{aligned} H(Y_n^{(m)}) &= - \sum_{\|k\| = n-m+1} \mathbb{P}(Y_n^{(m)} = k^{(m)}) \log \mathbb{P}(Y_n^{(m)} = k^{(m)}) \\ &= \frac{1}{\binom{n}{m-1}} \log \binom{n}{m-1} \sum_{\|k\| = n-m+1} 1 = \log \binom{n}{m-1}. \end{aligned}$$

The last equality comes from the fact that the sum $\sum_{\|k\| = n-m+1} 1$ equals to the number of choices of $m - 1$ pivots from n keys, which is $\binom{n}{m-1}$.

Finally, we arrive at the following recurrence for the entropy of unlabeled m -ary search trees:

$$H(U_n) = \log \binom{n}{m-1} + \frac{m}{\binom{n}{m-1}} \sum_{k=0}^{n-m+1} \binom{n-k-1}{m-2} H(U_k).$$

The asymptotics of a recurrence like this one have been studied before; see Proposition 7 in [4] and Theorem 2.4 in [12], which we quote below.

THEOREM 3.1 ([12], THEOREM 2.4, ASYMPTOTIC TRANSFER THEOREM). *Let*

$$a_n = b_n + \frac{m}{\binom{n}{m-1}} \sum_{j=0}^{n-(m-1)} \binom{n-1-j}{m-2} a_j, \quad n \geq m-1,$$

with specified initial conditions (say) $a_j := b_j$, $0 \leq j \leq m-2$. If

$$b_n = o(n) \quad \text{and} \quad \sum_{n \geq 0} \frac{b_n}{(n+1)(n+2)} \text{ converges,}$$

then

$$a_n = \frac{K_1}{\mathcal{H}_m - 1} n + o(n), \quad \text{where} \quad K_1 := \sum_{j \geq 0} \frac{b_j}{(j+1)(j+2)}.$$

Here, $\mathcal{H}_m$ is the m th harmonic number.

Remark 1. The asymptotics of our recurrence can also be derived using the *continuous master theorem* in [24], which is a rather more general tool.

Using Theorem 3.1 and the fact that $H(Y_n^{(m)}) = \log \binom{n}{m-1} = o(n)$, we conclude this section with the following result.

COROLLARY 3.2. *The entropy rate $h_m^u = \lim_{n \rightarrow \infty} H(U_n)/n$ of the unlabeled m -ary trees, generated according to the m -ary search tree model, is given by*

$$h_m^u = 2\phi_m \sum_{k \geq 0} \frac{\log \binom{k}{m-1}}{(k+1)(k+2)}, \quad (9)$$

where $\phi_m = \frac{1}{2H_{m-2}}$ is called the occupancy constant.

Remark 2. It can be checked numerically that the entropy rate for unlabeled m -ary search trees for $m = 2, 3, 4, 5$ is $h_2^u \approx 1.73638$, $h_3^u \approx 1.50084$, $h_4^u \approx 1.3569$, $h_5^u \approx 1.25723$, respectively. Notice that the entropy rate decreases as we increase m . Intuitively, this is to be expected: consider the extreme case, with $m = n$ (though, in the rest of the article, we always consider constant m). In this case, every pattern of insertions into an m -ary search tree results in the same unlabeled tree (since the m slots of the root are never filled), so the entropy is 0.

Remark 3. Observe that if $m = 2$ the number of nodes of an m -ary search tree equals n , the number of inserted keys; but for $m > 2$ the number of nodes is a random variable $S_{n,m}$. Knuth [19] was the first to show that $\mathbb{E}(S_{n,m}) \sim \phi_m n$.

In order to compare the entropy rate h_m^u of m -ary search trees with that for d -ary plane increasing trees (discussed next), it makes sense to normalize the constant h_m^u as $\hat{h}_m^u = h_m^u / \phi_m$. Then numerically $\hat{h}_2^u \approx 1.73638$, $\hat{h}_3^u \approx 2.5014$, $\hat{h}_4^u \approx 2.93994$, $\hat{h}_5^u \approx 3.22688$.

3.3 The Entropy of the Unlabeled d -ary Plane Increasing Trees

Let $g_n = |\mathcal{G}_n|$ be the number of d -ary plane increasing trees with n internal nodes. Trivially, for $d = 2$, we have $g_n = n!$. Moreover, for $d > 2$ we have

$$g_n = (-1)^n (d-1)^n \frac{\Gamma(2 - \frac{d}{d-1})}{\Gamma(2 - \frac{d}{d-1} - n)}, \quad (10)$$

which may be easily proven by induction.

Let $\mathcal{G}_{f_n}$ denote the subset of $\mathcal{G}_n$ of trees that have the same structure as the unlabeled tree $f_n \in \mathcal{F}_n$; that is, $\mathcal{G}_{f_n}$ is the set of labeled representatives of f_n . Moreover, let $g_{f_n} = |\mathcal{G}_{f_n}|$ be the number of d -ary plane increasing trees that have the same structure as a tree f_n . Observe that the probability that the d -ary plane increasing tree source generates a given unlabeled tree $f_n \in \mathcal{F}_n$ is

$$\mathbb{P}(F_n = f_n) = \frac{g_{f_n}}{g_n}. \quad (11)$$

Suppose that the tree f_n has subtrees $f_{k_1}, \dots, f_{k_d}$ of sizes $k_1, \dots, k_d$. Then

$$\mathbb{P}(F_n = f_n) = \frac{1}{g_n} \binom{n-1}{k_1, \dots, k_d} \prod_{j=1}^d g_{f_{k_j}} = \binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n} \prod_{j=1}^d \mathbb{P}(F_{k_j} = f_{k_j}). \quad (12)$$

Observe that $\binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n}$ is the probability that the subtrees of the root are of sizes $k_1, \dots, k_d$ and thus (12) can be also derived from Equation (2). Let us define a random vector $\mathbf{V}_n^{(d)} : \mathcal{G}_n \rightarrow \{0, \dots, n-1\}^d$ whose j th component $V_{n,j}$ denotes the size of the j th subtree. For $n \geq 1$ we have $V_{n,1} + \dots + V_{n,d} = n-1$ and

$$\mathbb{P}(\mathbf{V}_n^{(d)} = \mathbf{k}^{(d)}) = \binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n}. \quad (13)$$

Let us establish the initial conditions of the entropy of our source. If $n = 0$, we have an empty tree, and $H(F_0) = 0$. If $n = 1$, we have one fixed tree and $H(F_1) = 0$. By the definition of the trees, for $n > 1$ there is a bijection between a tree F_n and a tuple $(\mathbf{V}_n^{(d)}, F_{V_1}, \dots, F_{V_d})$. Therefore, for $n > 1$, we have

$$\begin{aligned} H(F_n) &= H(\mathbf{V}_n^{(d)}, F_{V_{n,1}}, \dots, F_{V_{n,d}}) = H(\mathbf{V}_n^{(d)}) + H(F_{V_{n,1}}, \dots, F_{V_{n,d}} | \mathbf{V}_n^{(d)}) \\ &= H(\mathbf{V}_n^{(d)}) + \sum_{\|\mathbf{k}\|=n-1} H(F_{k_1}, \dots, F_{k_d}) \mathbb{P}(\mathbf{V}_n^{(d)} = \mathbf{k}^{(d)}). \end{aligned}$$

Since subtrees $F_{k_1}, \dots, F_{k_d}$ are conditionally independent given their sizes, we have

$$H(F_n) = H(\mathbf{V}_n^{(d)}) + d \sum_{k=0}^{n-1} H(F_k) \sum_{\|\mathbf{k}^{(d-1)}\|=n-1-k} \mathbb{P}(\mathbf{V}_n^{(d)} = (k, \mathbf{k}^{(d-1)})).$$

For $k = 0, \dots, n-1$, let $p_{n,k}$ be the probability that one specified subtree in a d -ary plane increasing tree is of size k , that is,

$$p_{n,k} = \sum_{\|\mathbf{k}^{(d-1)}\|=n-1-k} \mathbb{P}(\mathbf{V}_n^{(d)} = (k, \mathbf{k}^{(d-1)})). \quad (14)$$

Then

$$H(F_n) = H(\mathbf{V}_n^{(d)}) + d \sum_{k=0}^{n-1} H(F_k) p_{n,k}. \quad (15)$$

The next lemma, proved in Section 4.1, gives an explicit formula for $p_{n,k}$.

LEMMA 3.3. For $k = 0, \dots, n-1$ and $d > 1$, let $\alpha = \frac{d}{d-1}$, then

$$p_{n,k} = \frac{(\alpha - 1)}{n} \frac{n! \Gamma(k + \alpha - 1)}{k! \Gamma(n + \alpha - 1)}.$$

Remark 4. Observe that for $d = 2$, we have $\alpha = 2$ and $p_{n,k} = \frac{1}{n}$. It does not depend on k , which greatly simplifies computations as shown in [21]. Moreover, it equals $\mathbb{P}(Y_{n,1} = k)$ in the case of the binary search trees. Therefore, the two models, one for binary plane increasing trees and the other for binary search trees, are equal. For $d > 2$, this is not the case. For instance, for $d = 3$, we have $\alpha = \frac{3}{2}$ and

$$p_{n,k} = \frac{1}{2n} \frac{\binom{2k}{k} 2^{2n}}{\binom{2n}{n} 2^{2k}},$$

which clearly depends on k and does not equal $\frac{n-k-1}{\binom{n}{2}}$, which would be the case for 3-ary search trees.

The recurrence presented in (15) is the one that we need to solve. In the lemma below, we propose a general solution for recurrences of this form. It is proved in Section 4.2.

LEMMA 3.4 (EXACT SOLUTION TO A GENERALIZED ENTROPY RECURRENCE FOR d -ARY PLANE INCREASING TREES). For constant α, x_0 and x_1 , the recurrence

$$x_n = a_n + \frac{\alpha}{n} \frac{n!}{\Gamma(n + \alpha - 1)} \sum_{k=0}^{n-1} \frac{\Gamma(k + \alpha - 1)}{k!} x_k, \quad n \geq 2 \quad (16)$$

Fig. 4. Entropy of d -ary plane increasing trees for $d = 2, 3, 4, 5$, respectively, and number of nodes $n \in [1, 100]$.

has the following solution for $n \geq 2$:

$$x_n = a_n + \alpha(n + \alpha - 1) \sum_{k=0}^{n-1} \frac{a_k}{(k + \alpha - 1)(k + \alpha)} + \frac{n + \alpha - 1}{\alpha + 1} \left(x_1 + \frac{x_0}{\alpha - 1} \right).$$

Remark 5. Observe again that for $d = 2$ and $x_0 = x_1 = 0$ and $a_n = o(n)$, we have $\alpha = 2$ and

$$x_n = 2n \sum_{k \geq 0} \frac{a_k}{(k+1)(k+2)} + o(n)$$

as in [21]. But with the same assumptions and $d = 3$ we have

$$x_n = \frac{3}{2}n \sum_{k \geq 0} \frac{a_k}{\left(k + \frac{1}{2}\right)\left(k + \frac{3}{2}\right)} + o(n).$$

This leads us to our first main result.

THEOREM 3.5. *The entropy of an unlabeled d -ary plane tree, generated according to the d -ary plane increasing tree model, is given by (see Figure 4)*

$$H(F_n) = H(\mathbf{V}_n^{(d)}) + \alpha(n + \alpha - 1) \sum_{k=0}^{n-1} \frac{H(\mathbf{V}_k^{(d)})}{(k + \alpha - 1)(k + \alpha)}, \quad (17)$$

where $\alpha = \frac{d}{d-1}$ and

$$H(\mathbf{V}_n^{(d)}) = - \sum_{\|\mathbf{k}\|=n-1} \mathbb{P}(\mathbf{V}_n^{(d)} = \mathbf{k}^{(d)}) \log \mathbb{P}(\mathbf{V}_n^{(d)} = \mathbf{k}^{(d)}).$$

We conclude this section with the following result.

COROLLARY 3.6. *The entropy rate $h_d^f = \lim_{n \rightarrow \infty} H(F_n)/n$ of the unlabeled d -ary plane trees, generated according to the model of d -ary plane increasing trees, is given by*

$$h_d^f = \alpha \sum_{k=0}^{\infty} \frac{H(\mathbf{V}_k)}{(k + \alpha - 1)(k + \alpha)}, \quad (18)$$

with $\alpha = \frac{d}{d-1}$.

PROOF. Having Theorem 3.5 in mind, we just need to prove that $H(\mathbf{V}_n^{(d)}) = o(n)$. Let us recall that $\mathbf{V}_n^{(d)} : \mathcal{G}_n \rightarrow \{0, \dots, n-1\}^d$. Since the entropy of random variable is upper bounded by the logarithm of the variable image cardinality, we have

$$H(\mathbf{V}_n^{(d)}) \leq \log(n^d) = o(n)$$

as needed. $\square$

Remark 6. Taking a closer look at $H(\mathbf{V}_n^{(d)})$, we find

$$H(\mathbf{V}_n^{(d)}) = \log\left(n \frac{g_n}{n!}\right) - d \sum_{k=0}^{n-1} p_{n,k} \log\left(\frac{g_k}{k!}\right).$$

In particular, $H(\mathbf{V}_n^{(2)}) = H(\mathbf{Y}_n^{(2)}) = \log(n-1)$ and the entropy rate $h_2^f \approx 1.73638$, which matches the entropy rate of the binary search trees. On the other hand, for $d=3$ and $n > 0$, we have

$$H(\mathbf{V}_n^{(3)}) = \log\left(\frac{n}{2^n} \binom{2n}{n}\right) - \frac{3}{2n} \sum_{k=0}^{n-1} \frac{\binom{2k}{k} 2^{2n}}{\binom{2n}{n} 2^{2k}} \log\left(\frac{\binom{2k}{k}}{2^k}\right).$$

This allows us to check numerically that the entropy rate of the unlabeled 3-ary plane trees, generated according to the model of 3-ary plane increasing trees is $h_3^f \approx 2.470$. It is worth noticing that the value h_3^f is smaller than $\hat{h}_3^u \approx 2.5014$ that is, the normalized entropy rate for 3-ary search trees.

3.4 The Entropy of the Unlabeled Degree-Unconstrained Plane Trees

Let $r_n = |\mathcal{R}_n|$, the number of labeled plane increasing trees with n nodes. We know that there are

$$r_n = (2n-3)!! = \frac{n!}{n2^{n-1}} \binom{2n-2}{n-1} \quad (19)$$

different labeled plane increasing trees of size n .

As in the case of the d -ary plane increasing trees, let $\mathcal{R}_{t_n}$ denote the subset of trees in $\mathcal{R}_n$ that have the same structure as a given unlabeled tree $t_n \in \mathcal{T}_n$ (i.e., $\mathcal{R}_{t_n}$ is the set of labeled *representatives* of t_n); moreover, let $r_{t_n} = |\mathcal{R}_{t_n}|$ be the number of such trees. Observe that

$$\mathbb{P}(T_n = t_n) = \frac{r_{t_n}}{r_n}. \quad (20)$$

Let D_n denote the random variable representing the number of subtrees of the root. Observe that $\mathbb{P}(D_n = d) = \frac{r_n^{(d)}}{r_n}$, where $r_n^{(d)} = |\mathcal{R}_n^{(d)}|$ is the number of plane increasing trees with root degree equal to d . Suppose that the tree t_n has d subtrees $t_{k_1}, \dots, t_{k_d}$ of sizes $k_1, \dots, k_d$. Then by (2) and the fact that $\mathbb{P}(T_n = t_n) = \mathbb{P}(T_n = t_n, D_n = d)$ by assumption,

$$\mathbb{P}(T_n = t_n) = \mathbb{P}(D_n = d) \mathbb{P}(T_n = t_n | D_n = d) = \binom{n-1}{k_1, \dots, k_d} \frac{r_{k_1} \cdots r_{k_d}}{r_n} \prod_{j=1}^d \mathbb{P}(T_{k_j} = t_{k_j}). \quad (21)$$

Observe that $\binom{n-1}{k_1, \dots, k_d} \frac{r_{k_1} \cdots r_{k_d}}{r_n}$ is the probability that the root of a plane increasing tree of size n has degree equal to d and the root's subtrees are of sizes $k_1, \dots, k_d$. Let $\mathbf{W}_n^{(d)} : \mathcal{R}_n^{(d)} \rightarrow \{1, \dots, n-d\}^d$, where its j th component $W_{n,j}$ denotes the size of the j th subtree when the root is of degree d . For $n \geq 1$ we have $W_{n,1} + \dots + W_{n,d} = n-1$ and

$$\mathbb{P}(D_n = d) \mathbb{P}(\mathbf{W}_n^{(D_n)} = \mathbf{k}^{(D_n)} | D_n = d) = \binom{n-1}{k_1, \dots, k_d} \frac{r_{k_1} \cdots r_{k_d}}{r_n}. \quad (22)$$

The initial conditions for the entropy are as follows. If $n = 1$, we have just a root node, so $H(T_1) = 0$. Similarly, if $n = 2$, we have one fixed tree, so $H(T_2) = 0$. We now give the recurrence for $n > 2$. For $k = 1, \dots, n-1$, let $q_{n,k}^{(d)}$ be defined as the probability that the root of a plane increasing tree has degree d and that one specified root subtree is of size k . Then

$$q_{n,k}^{(d)} = \mathbb{P}(D_n = d) \sum_{\|\mathbf{k}^{(d-1)}\| = n-1-k} \mathbb{P}(\mathbf{W}_n^{(d)} = (k, \mathbf{k}^{(d-1)})) . \quad (23)$$

Therefore, largely by the same reasoning as was used to derive the recurrence for d -ary plane trees,

$$H(T_n) = \sum_{d=1}^{n-1} H(\mathbf{W}_n^{(d)} | D_n = d) \mathbb{P}(D_n = d) + \sum_{d=1}^{n-1} d \sum_{k=1}^{n-d} H(T_k) q_{n,k}^{(d)} . \quad (24)$$

We need an expression for the probability $q_{n,k}^{(d)}$ which we present in the next lemma proved in Section 4.3.

LEMMA 3.7. *For $k = 1, \dots, n-1$, we have*

$$\begin{aligned} -q_{n,n-1}^{(1)} &= \frac{1}{2n-3} \text{ and if } k \neq n-1 : q_{n,k}^{(1)} = 0, \\ -\text{for } d > 1: \end{aligned}$$

$$q_{n,k}^{(d)} = 2^d \frac{d-1}{k(n-1-k)} \frac{\binom{2k-2}{k-1} \binom{2(n-1-k)-d}{n-2-k}}{\binom{2n-2}{n-1}} .$$

The recurrence found in (24) is another one that we need to analyze. Its general solution is presented next and proved in Section 4.4.

LEMMA 3.8 (EXACT SOLUTION TO A GENERALIZED ENTROPY RECURRENCE FOR UNRESTRICTED TREES). *For constant y_1 and y_2 , the recurrence*

$$y_n = b_n + \sum_{d=1}^{n-1} d \sum_{k=1}^{n-d} q_{n,k}^{(d)} \cdot y_k, \quad n > 2 \quad (25)$$

has the following solution for $n > 2$:

$$y_n = \frac{2(2n-1)}{3} b_1 + b_n + \frac{1}{2} \left(n - \frac{1}{2} \right) \sum_{j=2}^{n-1} \frac{b_j}{\left(j - \frac{1}{2} \right) \left(j + \frac{1}{2} \right)} .$$

This leads us to our second main result.

THEOREM 3.9. *The entropy of an unlabeled degree-unconstrained plane tree, generated according to the model of plane increasing tree, is given by*

$$H(T_n) = \sum_{d=1}^{n-1} H(\mathbf{W}_n^{(d)} | D_n = d) \mathbb{P}(D_n = d) + \frac{1}{2} \left(n - \frac{1}{2} \right) \sum_{j=2}^{n-1} \frac{\sum_{d=1}^{j-1} H(\mathbf{W}_j^{(d)} | D_j = d) \mathbb{P}(D_j = d)}{\left(j - \frac{1}{2} \right) \left(j + \frac{1}{2} \right)} , \quad (26)$$

where

$$H(\mathbf{W}_n^{(d)} | D_n = d) = - \sum_{\|\mathbf{k}\| = n-1} \mathbb{P}(\mathbf{W}_n^{(d)} = \mathbf{k}^{(d)} | D_n = d) \log \mathbb{P}(\mathbf{W}_n^{(d)} = \mathbf{k}^{(d)} | D_n = d) .$$

We conclude this section with the following result.

COROLLARY 3.10. *The entropy rate $h_t = \lim_{n \rightarrow \infty} H(T_n)/n$ of the unlabeled degree-unconstrained plane trees, generated according to the model of plane increasing trees, is given by*

$$h_t = \frac{1}{2} \sum_{j=2}^{\infty} \frac{\sum_{d=1}^{j-1} H(\mathbf{W}_k^{(d)} | D_n = d) \mathbb{P}(D_n = d)}{(j - \frac{1}{2})(j + \frac{1}{2})}. \quad (27)$$

PROOF. From Theorem 3.9, we just need to prove that $\sum_{d=1}^{n-1} H(\mathbf{W}_n^{(d)} | D_n = d) \mathbb{P}(D_n = d) = o(n)$. Let us recall that the random vector $\mathbf{W}_n^{(d)} : \mathcal{R}_n^{(d)} \rightarrow \{1, \dots, n-d\}^d$ describes the split at the root of a tree: precisely that a tree root degree equals d and its subtrees are of sizes $W_{n,1}^{(d)}, \dots, W_{n,d}^{(d)}$. Since the entropy of random variable is upper bounded by the logarithm of the variable image cardinality, we have

$$\sum_{d=1}^{n-1} \mathbb{P}(D_n = d) H(\mathbf{W}_n^{(D_n)} | D_n = d) \leq \sum_{d=1}^{n-1} \mathbb{P}(D_n = d) \log(n^d) = \log(n) \sum_{d=1}^{n-1} d \mathbb{P}(D_n = d).$$

Observe that $\mathbb{E}(D_n) = \sum_{d=1}^{n-1} d \mathbb{P}(D_n = d)$ is the expected value of the degree-unconstrained plane increasing tree root degree. From [2] we know that $\mathbb{E}(D_n) = \sqrt{\pi n} + O(1)$, which gives us the desired result. $\square$

Remark 7. Taking a closer look at $H(\mathbf{W}_n^{(D_n)} | D_n)$, we find that

$$H(\mathbf{W}_n^{(D_n)} | D_n) = \log \binom{n}{n!} - \sum_{d=1}^{n-1} d \sum_{k=1}^{n-d} q_{n,k}^{(d)} \log \left(\frac{r_k}{k!} \right).$$

This allows us to check numerically that the entropy rate of the unlabeled degree-unconstrained plane trees, generated according to the model of plane increasing trees is $h_t \approx 1.68$.

3.5 Optimal Compression

Here we address optimal compression of the trees generated by the sources under consideration. We will use a variant of the arithmetic coding method. We explain in detail the generalization for compression of unlabeled d -ary plane increasing trees (the case of degree-unconstrained plane increasing trees can be handled along analogous lines).

In a nutshell, we first define a total order $<$ on the set of such trees with a given size. Having fixed this, we must show how to efficiently compute two quantities, for a given tree t : the probability of all trees $t' < t$, as well as the probability of t itself. Granted efficient procedures for computing these, we produce a subinterval $I(t)$ of $[0, 1]$, unique to t , which has length $|I(t)|$ equal to the probability of t and whose left endpoint is the probability of all trees $t' < t$. Arithmetic coding then prescribes that the code word corresponding to t be given by the binary expansion of the midpoint of the interval assigned to t , truncated to a length of $\lceil \log |I(t)| \rceil + 1$ bits. The expected length of this code is then easily at most the entropy of the source, plus at most 2 bits.

We now define the total order on the set $\mathcal{F}_n$ on unlabeled d -ary trees: we denote by $<$ the lexicographic order on tuples of non-negative integers, and then we have the following definition.

Definition 3.11 (Total Order of the Set of Unlabeled d -ary Plane Trees). The relation $<$ on $\mathcal{F}$ is defined as follows: let $f_1, f_2 \in \mathcal{F}$ with subtrees sizes $(s_1, \dots, s_d), (k_1, \dots, k_d)$ respectively, then $f_1 < f_2$ if and only if one of the following holds:

- $(s_1, \dots, s_d) < (k_1, \dots, k_d)$,
- or if $(s_1, \dots, s_d) = (k_1, \dots, k_d)$ and first subtree of $f_1 <$ first subtree of f_2 ,
- or if $(s_1, \dots, s_d) = (k_1, \dots, k_d)$, first subtree of $f_1 =$ first subtree of f_2 and second subtree of $f_1 <$ second subtree of f_2 ,
- ...
- or if $(s_1, \dots, s_d) = (k_1, \dots, k_d)$, first $d - 1$ subtrees of $f_1 =$ first $d - 1$ subtrees of f_2 and d th subtree of $f_1 <$ d th subtree of f_2 .

It is simple to check that this is a total order. Next, we present an algorithm which computes the subinterval corresponding to an input tree $f \in \mathcal{F}$ (see Algorithm 1).

ALGORITHM 1: Unlabeled d -ary Plane Tree Compression

```

1: function COMPRESSDTREE( $f \in \mathcal{F}$ ) ▷  $f$  tree to be compressed
2:    $[a, b] \leftarrow \text{EXPLORE}(\text{root of } f, [0, 1])$ 
3:   return first  $\lceil -\log_2(b - a) \rceil + 1$  bits of  $(a + b)/2$ 

4: function EXPLORE( $v \in f, [l, r] \subseteq [0, 1]$ )
5:    $\text{visited}(v) \leftarrow \text{true}$ 
6:    $n \leftarrow$  size of a subtree of  $f$  hanging from node  $v$ 
7:    $(s_1, \dots, s_d) \leftarrow$  sizes of subtrees of  $v$ 
8:    $a \leftarrow l + (r - l) \cdot \text{CALCULATEINTERVALBEGIN}(n, s_1, \dots, s_d)$ 
9:    $p \leftarrow (r - l) \cdot \text{CALCULATESPITPROBABILITY}(n, s_1, \dots, s_d)$ 
10:   $I_{\text{new}} \leftarrow [a, a + p]$ 
11:  for all  $u$  descendant of  $v$  do
12:    if not  $\text{visited}(u)$  then
13:       $I_{\text{new}} \leftarrow \text{EXPLORE}(u, I_{\text{new}})$ 
14:  return  $I_{\text{new}}$ 

15: function CALCULATESPITPROBABILITY( $n, k_1, \dots, k_d$ )
16:  return  $\binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \dots g_{k_d}}{g_n}$ 

17: function CALCULATEINTERVALBEGIN( $n, s_1, \dots, s_d$ )
18:  return
    
$$\frac{1}{g_n} \sum_{i=1}^d \frac{(n-1)!}{s_1! \dots s_{i-1}!} \left( \prod_{j=1}^{i-1} g_{s_j} \right) \sum_{k=0}^{s_i-1} \frac{g_k}{k!} \left( n - 2 - k - \sum_{l=1}^{i-1} s_l + \frac{d-i}{d-1} \right) (d-1)^{n-1-k-\sum_{l=1}^{i-1} s_l} \frac{1-i}{d-1}$$


```

This does exactly as intuitively described above: it implements a depth-first search of the input tree f , and at each step refining the current interval based on the split of vertices among the root subtrees of the current node.

Now, we explain more precisely the procedures CALCULATESPITPROBABILITY and CALCULATEINTERVALBEGIN. The former simply calculates the probability that a d -ary tree of size n has root subtrees of sizes $k_1, \dots, k_d$ (giving the length of the next subinterval). This is illustrated in Figure 5.

Fig. 5. Visualization of lines 8–10 in the Algorithm 1 listing.

The latter gives the probability that such a d -ary tree has a root subtree size tuple lexicographically less than $(s_1, \dots, s_d)$. That is, it computes the expression

$$\sum_{\substack{(k_1, \dots, k_d) < (s_1, \dots, s_d) \\ k_1 + \dots + k_d = n-1}} \binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n}.$$

Observe that a naive implementation of this calculation generates all $\Theta(n^d)$ integer partitions with d parts of the number $n-1$ and calculates the split probability for each of them. To reduce the time complexity, we rewrite the sum as follows:

$$\begin{aligned} & \sum_{\substack{(k_1, \dots, k_d) < (s_1, \dots, s_d) \\ k_1 + \dots + k_d = n-1}} \binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n} \\ &= \frac{1}{g_n} \sum_{i=1}^d g_{s_1} \cdots g_{s_{i-1}} \sum_{k=0}^{s_i-1} \sum_{j_{i+1}+\dots+j_d=k} \binom{n-1}{s_1, \dots, s_{i-1}, k, j_{i+1}, \dots, j_d} g_k g_{j_{i+1}} \cdots g_{j_d}. \end{aligned}$$

For a given i , the i th term of the outermost sum gives the contribution of all tuples of the form $(s_1, \dots, s_{i-1}, k_i, k_{i+1}, \dots, k_d)$ with varying $k_i, \dots, k_d$. We can, furthermore, write the multinomial coefficient as a product of two other multinomial coefficients, one of which can be brought outside the k sum. The k th term of the resulting sum can then be written as follows:

$$\begin{aligned} & \frac{1}{(n-1-k-\sum_{l=1}^{i-1} s_l)!} \sum_{j_{i+1}+\dots+j_d=k} \binom{n-1-k-\sum_{l=1}^{i-1} s_l}{j_{i+1}, \dots, j_d} g_{j_{i+1}} \cdots g_{j_d} \\ &= \left[z^{n-1-k-\sum_{l=1}^{i-1} s_l} \right] (1 - (d-1)z)^{-\frac{d-i}{d-1}} \\ &= \binom{n-2-k-\sum_{l=1}^{i-1} s_l + \frac{d-i}{d-1}}{\frac{1-i}{d-1}} (d-1)^{n-1-k-\sum_{l=1}^{i-1} s_l}. \end{aligned}$$

Fig. 6. Illustration to Algorithm 1.

The following example describes algorithm COMPRESSDTREE in detail for a given 3-ary tree. From Equation (12), probability of the given tree equals to $\frac{4!}{9 \cdot 4 \cdot 1 \cdot 2 \cdot 1} = \frac{1}{35}$. Moreover, consecutive calls of the EXPLORE procedure for the tree vertices outputs the following intervals: $v_1 \rightarrow [\frac{1}{7}, \frac{8}{35})$, $v_2 \rightarrow [\frac{1}{7}, \frac{8}{35})$, $v_3 \rightarrow [\frac{6}{35}, \frac{1}{5})$, $v_4 \rightarrow [\frac{6}{35}, \frac{1}{5})$. In the last step $\lceil -\log_2 (\frac{1}{5} - \frac{6}{35}) \rceil + 1 = 7$ bits of the $\frac{1}{2} (\frac{6}{35} + \frac{1}{5}) = \frac{13}{70} = 0.001011111000101011111\dots$ is returned, i.e. 0010111.

We thus get, finally,

$$\sum_{\substack{(k_1, \dots, k_d) < (s_1, \dots, s_d) \\ k_1 + \dots + k_d = n-1}} \binom{n-1}{k_1, \dots, k_d} \frac{g_{k_1} \cdots g_{k_d}}{g_n} \\ = \frac{1}{g_n} \sum_{i=1}^d \frac{(n-1)!}{s_1! \cdots s_{i-1}!} \left(\prod_{j=1}^{i-1} g_{s_j} \right) \sum_{k=0}^{s_i-1} \frac{g_k}{k!} \binom{n-2-k-\sum_{l=1}^{i-1} s_l + \frac{d-i}{d-1}}{\frac{1-i}{d-1}} (d-1)^{n-1-k-\sum_{l=1}^{i-1} s_l}.$$

Observe that the resulting expression only requires one to perform $O(n)$ calculations, where each of them is of the same order as the calculation of the split probability. An application of Algorithm 1 to a 3-tree is presented in Figure 6.

This leads us to the time complexity of the algorithm: observe that for each vertex v of an input tree f of size n , the procedure EXPLORE calls one time both procedures CALCULATESPLITPROBABILITY and CALCULATEINTERVALBEGIN. Therefore, we get $O(n)$ calls of both procedures. Since the CALCULATEINTERVALBEGIN procedure performs $O(n)$ calculations where each of them is of the same time complexity as one CALCULATESPLITPROBABILITY procedure, we have that the compression algorithm runs in time $O(n^2 \cdot f(d, n))$, where $f(d, n)$ denotes the number of bit operations in the CALCULATESPLITPROBABILITY procedure (it is not too difficult to see that $f(d, n)$ can be bounded by a small polynomial function of n alone, by taking into account cancellations of factors in the expression for the split probabilities).

Finally, standard arithmetic coding arguments show that the algorithm is *optimal* up to a small constant number of bits, in expectation. Experimental results confirming this statement can be seen in Table 1.

Remark 8. It is instructive to compare the performance of our algorithm with a natural compact representation (which we think of as an uncompressed representation) of the trees in question. In particular, in the *parenthesis representation*, a plane tree is encoded as a bit string via a depth-first search. When a node is first encountered (and pushed), a 0 is appended. When the same node is popped (i.e., the search leaves that node's subtree), a 1 is appended.

In such a representation, the number of bits needed per vertex of a d -ary tree is approximately $2d$. In Figure 7 we highlight the gain obtained by using our optimal compression algorithm. Although we observe that $h_d^f = \Theta(d)$, the difference in constant is significant.

Table 1. Comparison of the Entropy of d -ary Plane Increasing Trees and the Compressed Code Length Obtained from Experimental Results (100 Uniformly Drawn Trees for Each d and n Sizes of a Tree)

n	50	100	150	200	250	300	350	400	450	500
$d = 3$										
$H(F_n)$	115.63	238.17	361.12	484.24	607.46	730.73	854.04	977.39	1,100.75	1,224.14
code length	117.04	239.46	362.07	484.94	608.70	731.56	853.72	978.67	1,103.61	1,223.66
$d = 8$										
$H(F_n)$	192.63	394.08	595.95	797.99	1,000.12	1,202.31	1,404.54	1,606.80	1,809.09	2,011.39
code length	193.85	396.05	599.79	799.56	1,001.24	1,205.07	1,404.22	1,609.57	1,810.42	2,012.90

Fig. 7. Entropy rate of d -ary plane increasing trees and the number of bits needed to encode one vertex of d -ary tree in the parenthesis representation.

4 PROOFS OF TECHNICAL LEMMAS

4.1 Proof of Lemma 3.3

Using (13), we can rewrite (14) as

$$p_{n,k} = \frac{(n-1)!g_k}{k!(n-1-k)!g_n} \sum_{k_2+\dots+k_d=n-1-k} \binom{n-1-k}{k_2, \dots, k_d} g_{k_2} \cdots g_{k_d}.$$

Let us define the exponential generating function $G(z) = \sum_{n \geq 0} g_n \frac{z^n}{n!}$ with $g_0 = 1$. From [10] we know that

$$G(z) = (1 - (d-1)z)^{-\frac{1}{d-1}}.$$

Observe that

$$\sum_{k_2+\dots+k_d=n-1-k} \binom{n-1-k}{k_2, \dots, k_d} g_{k_2} \cdots g_{k_d}$$

is the $n - 1 - k$ th coefficient of the function $G(z)^{d-1}$ (denoted as $\left[\frac{z^{n-1-k}}{(n-1-k)!}\right] G(z)^{d-1}$). Hence,

$$\begin{aligned} p_{n,k} &= \frac{(n-1)!g_k}{k!(n-1-k)!g_n} \left[\frac{z^{n-1-k}}{(n-1-k)!} \right] G(z)^{d-1} = \frac{(n-1)!g_k}{k!g_n} [z^{n-1-k}] \frac{1}{1-(d-1)z} \\ &= \frac{(n-1)!g_k}{k!g_n} (d-1)^{n-1-k}. \end{aligned}$$

For $d = 2$, we have $g_n = n!$ and the result is immediate. For $d > 2$, from (10) we find

$$p_{n,k} = \frac{(\alpha-1)}{n} \frac{(-1)^n n! \Gamma(2-\alpha-n)}{(-1)^k k! \Gamma(2-\alpha-k)}.$$

From [11] we know that $\Gamma(z-n) = \frac{(-1)^n \pi}{\Gamma(n+1-z) \sin(\pi z)}$; hence,

$$(-1)^n \Gamma(n+\alpha-1) \Gamma(2-\alpha-n) = \frac{\pi}{\sin(\pi(2-\alpha))}, \quad (28)$$

and then

$$p_{n,k} = \frac{(\alpha-1)}{n} \frac{n! \Gamma(k+\alpha-1)}{k! \Gamma(n+\alpha-1)},$$

which gives the desired result.

4.2 Proof of Lemma 3.4

Let us multiply both sides of the recurrence by the normalizing factor $\frac{\Gamma(n+\alpha-1)}{n!}$. Define also

$$\hat{x}_n = \frac{x_n \Gamma(n+\alpha-1)}{n!}, \quad \hat{a}_n = \frac{a_n \Gamma(n+\alpha-1)}{n!}.$$

Then

$$\hat{x}_n = \hat{a}_n + \frac{\alpha}{n} \sum_{k=2}^{n-1} \hat{x}_k. \quad (29)$$

To solve the recurrence (29) we compute $n\hat{x}_n - (n-1)\hat{x}_{n-1}$. This leads us to

$$\hat{x}_n = \hat{a}_n - \left(1 - \frac{1}{n}\right) \hat{a}_{n-1} + \left(1 + \frac{\alpha-1}{n}\right) \hat{x}_{n-1},$$

which holds for $n \geq 3$. Then after iterating the above we arrive at

$$\hat{x}_n = \hat{x}_2 \prod_{j=3}^n \left(1 + \frac{\alpha-1}{j}\right) + \sum_{k=3}^n \left(\hat{a}_k - \left(1 - \frac{1}{k}\right) \hat{a}_{k-1}\right) \prod_{j=k+1}^n \left(1 + \frac{\alpha-1}{j}\right). \quad (30)$$

The product $\prod_{j=k+1}^n \left(1 + \frac{\alpha-1}{j}\right) = \frac{k! \Gamma(n+\alpha)}{n! \Gamma(k+\alpha)}$, and after some standard calculations we obtain

$$\hat{x}_n = \hat{a}_n + (\hat{x}_2 - \hat{a}_2) \frac{2\Gamma(n+\alpha)}{\Gamma(\alpha+2)n!} + \frac{\Gamma(n+\alpha)}{n!} \sum_{k=2}^{n-1} \hat{a}_k \frac{k!}{\Gamma(k+\alpha)} \frac{\alpha}{k+\alpha}.$$

Going back from $\hat{x}_n$ and $\hat{a}_n$ to x_n, a_n , respectively, we obtain

$$x_n = a_n + \alpha(n+\alpha-1) \sum_{k=2}^{n-1} \frac{a_k}{(k+\alpha-1)(k+\alpha)} + (x_2 - a_2) \frac{n+\alpha-1}{\alpha+1}.$$

Observe that $x_2 - a_2 = x_1 + \frac{x_0}{\alpha-1}$. This completes the proof.

4.3 Proof of Lemma 3.7

If $d = 1$, then the root has only 1 subtree with all other nodes, so its size has to be equal to $n - 1$ and

$$q_{n,n-1}^{(1)} = \frac{r_{n-1}}{r_n} = \frac{1}{2n-3};$$

moreover, if $k \neq n - 1$: $q_{n,k}^{(1)} = 0$. In the case of $d > 1$, using (22), we can rewrite (23) as follows:

$$q_{n,k}^{(d)} = \frac{(n-1)!r_k}{k!(n-1-k)!r_n} \times \sum_{k_2+\dots+k_d=n-1-k} \binom{n-1-k}{k_2, \dots, k_d} r_{k_2} \cdots r_{k_d}.$$

Let us define the exponential generating function $R(z) = \sum_{n \geq 0} r_n \frac{z^n}{n!}$ with $g_0 = 0$. Observe that

$$\sum_{k_2+\dots+k_d=n-1-k} \binom{n-1-k}{k_2, \dots, k_d} r_{k_2} \cdots r_{k_d}$$

is the $n - 1 - k$ th coefficient of the function $R(z)^{d-1}$ (denoted as $\left[\frac{z^{n-1-k}}{(n-1-k)!} \right] R(z)^{d-1}$). Therefore,

$$q_{n,k} = \frac{(n-1)!r_k}{k!r_n} [z^{n-1-k}] R(z)^{d-1}.$$

From (19) we find $R(z) = 1 - \sqrt{1 - 2z}$, which is also the solution of the equation

$$R = \frac{z}{1 - \frac{R}{2}}.$$

Hence, by Lagrange's inversion formula (see [14]), we obtain an explicit formula for

$$[z^{n-1-k}] R(z)^{d-1} = 2^{d-n+k} \frac{d-1}{n-1-k} \binom{2(n-1-k)-d}{n-2-k}.$$

Putting everything together, we arrive at the desired result.

4.4 Proof of Lemma 3.8

Using Lemma 3.7, for $n > 2$, we find

$$y_n = b_n + \frac{y_{n-1}}{2n-3} + \sum_{d=2}^{n-1} d(d-1)2^d \sum_{k=1}^{n-d} \frac{y_k}{k(n-1-k)} \frac{\binom{2k-2}{k-1} \binom{2n-2k-2-d}{n-k-2}}{\binom{2n-2}{n-1}}.$$

Multiplying both sides by $\frac{\binom{2n-2}{n-1}}{n}$ and substituting $\hat{y}_n = \frac{y_n \binom{2n-2}{n-1}}{n}$, $\hat{b}_n = \frac{b_n \binom{2n-2}{n-1}}{n}$ we get

$$\hat{y}_n = \hat{b}_n + \frac{2\hat{y}_{n-1}}{n(n-1)} + \frac{1}{n} \sum_{d=2}^{n-1} d(d-1)2^d \sum_{k=1}^{n-d} \frac{\hat{y}_k}{(n-1-k)} \binom{2n-2k-2-d}{n-k-2}.$$

Changing the order of summation gives us

$$\sum_{d=2}^{n-1} d(d-1)2^d \sum_{k=1}^{n-d} \frac{\hat{y}_k}{(n-1-k)} \binom{2n-2k-2-d}{n-k-2} = \sum_{j=1}^{n-2} \frac{\hat{y}_j}{n-j-1} \sum_{s=0}^{n-j} s(s-1)2^s \binom{2n-2j-2-s}{n-j-2}.$$

Since for $N > 0$:

$$\sum_{s=0}^N s(s-1)2^s \binom{2N-2-s}{N-2} = (N-1)2^{2N-1},$$

we obtain

$$\hat{y}_n = \hat{b}_n + \frac{2\hat{y}_{n-1}}{n(n-1)} + \frac{1}{n} \sum_{j=1}^{n-2} \hat{y}_j 2^{2n-2j-1}.$$

Dividing both sides by 2^{2n} and substituting $\tilde{y}_n = \frac{\hat{y}_n}{2^{2n}}$, $\tilde{b}_n = \frac{\hat{b}_n}{2^{2n}}$ we find

$$\tilde{y}_n = \tilde{b}_n + \frac{1}{2n} \sum_{j=1}^{n-1} \tilde{y}_j.$$

Solving this recurrence relation by calculating $n\tilde{y}_n - (n-1)\tilde{y}_{n-1}$, we obtain

$$\tilde{y}_n = b_1 \frac{\Gamma\left(n + \frac{1}{2}\right)}{\Gamma\left(\frac{5}{2}\right) n!} + \tilde{b}_n + \frac{\Gamma\left(n + \frac{1}{2}\right)}{n!} \sum_{j=2}^{n-1} \frac{\tilde{b}_j}{2j+1} \frac{j!}{\Gamma\left(j + \frac{1}{2}\right)}.$$

Substituting $\tilde{y}_n$ into y_n with $\tilde{y}_n = y_n \frac{\binom{2n-2}{n-1}}{n^{2n}}$, we find the desired result.

5 CONCLUDING REMARKS

In this article, we focused on finding entropies of various trees: namely, m -ary search trees, d -ary plane increasing trees, and degree-unconstrained plane increasing trees. In the course of deriving these entropies, we encountered recurrences that we showed how to solve in a general setting. These recurrences find ample applications in analyzing variations on such general trees. For example, as in [21], the next natural question is to find the entropy of non-plane d -ary increasing trees and degree-unconstrained trees. For arbitrary d , such a problem is quite challenging as one can see the approach of [6] for the binary case. Generalization to the d -ary case is highly non-trivial.

Finally, we presented optimal, polynomial-time (with small exponents) compression algorithms that achieve these entropies via an arithmetic coding approach.

REFERENCES

- [1] David J. Aldous and Nathan Ross. 2014. Entropy of some models of sparse random graphs with vertex-names. *Probability in the Engineering and Information Sciences* 28, 2 (2014), 145–168. DOI : <https://doi.org/10.1017/S0269964813000399>
- [2] François Bergeron, Philippe Flajolet, and Bruno Salvy. 1992. *Varieties of Increasing Trees*. Springer, Berlin, 24–48. DOI : https://doi.org/10.1007/3-540-55251-0_2
- [3] Hua-Huai Chern, María-Inés Fernández-Camacho, Hsien-Kuei Hwang, and Conrado Martínez. 2014. Psi-series method for equality of random trees and quadratic convolution recurrences. *Random Structures & Algorithms* 44, 1 (2014), 67–108. DOI : <https://doi.org/10.1002/rsa.20428>
- [4] Hua-Huai Chern and Hsien-Kuei Hwang. 2001. Phase changes in random m -ary search trees and generalized quick-sort. *Random Structures & Algorithms* 19, 3–4 (2001), 316–358. DOI : <https://doi.org/10.1002/rsa.10005>
- [5] Yongwook Choi and Wojciech Szpankowski. 2012. Compression of graphical structures: Fundamental limits, algorithms, and experiments. *IEEE Transactions on Information Theory* 58, 2 (Feb. 2012), 620–638. DOI : <https://doi.org/10.1109/TIT.2011.2173710>
- [6] Jacek Cichoń, Abram Magner, Wojciech Szpankowski, and Krzysztof Turowski. 2017. *On Symmetries of Non-plane Trees in a Non-uniform Model*. In *2017 Proceedings of the 14th Workshop on Analytic Algorithmics and Combinatorics (ANALCO'17)*. 156–163. DOI : <https://doi.org/10.1137/1.9781611974775.16> arXiv:<http://epubs.siam.org/doi/pdf/10.1137/1.9781611974775.16>
- [7] Thomas Cover and Joy Thomas. 2006. *Elements of Information Theory* (2nd ed.). John Wiley & Sons.
- [8] Alexis Darrasse, Hsien-Kuei Hwang, and Michèle Soria. 2016. Shape measures of random increasing k -trees. *Combinatorics, Probability and Computing* 25, 5 (2016), 668–699. DOI : <https://doi.org/10.1017/S0963548316000018>
- [9] Michael Drmota. 2009. The height of increasing trees. *Annals of Combinatorics* 12, 4 (Apr. 2009), 373–402. DOI : <https://doi.org/10.1007/s00026-009-0009-x>
- [10] Michael Drmota. 2009. *Random Trees: An Interplay Between Combinatorics and Probability* (1st ed.). Springer Publishing Company, Incorporated.
- [11] Daniel Zwillinger (Ed.). 2011. *CRC Standard Mathematical Tables and Formulae*. CRC Press, Boca Raton, FL.

- [12] James Allen Fill and Nevin Kapur. 2005. Transfer theorems and asymptotic distributional results for m -ary search trees. *Random Structures & Algorithms* 26, 4 (July 2005), 359–391. DOI : <https://doi.org/10.1002/rsa.v26:4>
- [13] Philippe Flajolet, Xavier Gourdon, and Conrado Martínez. 1997. Patterns in random binary search trees. *Random Structures & Algorithms* 11, 3 (1997), 223–244. DOI : [https://doi.org/10.1002/\(SICI\)1098-2418\(199710\)11:3<223::AID-RSA2>3.0.CO;2-2](https://doi.org/10.1002/(SICI)1098-2418(199710)11:3<223::AID-RSA2>3.0.CO;2-2)
- [14] Philippe Flajolet and Robert Sedgewick. 2009. *Analytic Combinatorics* (1st ed.). Cambridge University Press, New York, NY.
- [15] Zbigniew Golebiewski, Marcin Kardas, Jakub Lemiesz, and Krzysztof Majcher. 2017. On structural entropy of uniform random intersection graphs. In *2017 IEEE International Symposium on Information Theory (ISIT'17)*. 1470–1474. DOI : <https://doi.org/10.1109/ISIT.2017.8006773>
- [16] Zbigniew Golebiewski, Abram Magner, and Wojciech Szpankowski. 2017. Entropy of some general plane trees. In *2017 IEEE International Symposium on Information Theory (ISIT'17)*. 301–305. DOI : <https://doi.org/10.1109/ISIT.2017.8006538>
- [17] Svante Janson, Markus Kuba, and Alois Panholzer. 2011. Generalized Stirling permutations, families of increasing trees and urn models. *Journal of Combinatorial Theory, Series A* 118, 1 (2011), 94–114. DOI : <https://doi.org/10.1016/j.jcta.2009.11.006>
- [18] John C. Kieffer, En hui Yang, and Wojciech Szpankowski. 2009. Structural complexity of random binary trees. In *2009 IEEE International Symposium on Information Theory*. 635–639. DOI : <https://doi.org/10.1109/ISIT.2009.5205704>
- [19] Donald E. Knuth. 1998. *The Art of Computer Programming, Volume 3: Sorting and Searching* (2nd ed.). Addison Wesley Longman Publishing Co., Inc., Redwood City, CA.
- [20] Tomasz Łuczak, Abram Magner, and Wojciech Szpankowski. 2017. Structural information and compression of scale-free graphs. *Preprint*.
- [21] Abram Magner, Krzysztof Turowski, and Wojciech Szpankowski. 2016. Lossless compression of binary trees with correlated vertex names. In *2016 IEEE International Symposium on Information Theory (ISIT'16)*. 1217–1221. DOI : <https://doi.org/10.1109/ISIT.2016.7541492>
- [22] Mehryar Mohri, Michael Riley, and Ananda Theertha Suresh. 2015. Automata and graph compression. In *2015 IEEE International Symposium on Information Theory (ISIT'15)*. 2989–2993. DOI : <https://doi.org/10.1109/ISIT.2015.7283005>
- [23] Dimbinaina Ralaivaosaona and Stephan Wagner. 2016. Additive functionals of d -ary increasing trees. In *Proceedings of the 27th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms*. 297–308.
- [24] Salvador Roura. 2001. Improved master theorems for divide-and-conquer recurrences. *Journal of the ACM* 48, 2 (March 2001), 170–205. DOI : <https://doi.org/10.1145/375827.375837>
- [25] Stephan Wagner. 2015. Central limit theorems for additive tree parameters with small toll functions. *Combinatorics, Probability and Computing* 24, 1 (2015), 329–353. DOI : <https://doi.org/10.1017/S0963548314000443>
- [26] Jie Zhang, En hui Yang, and John C. Kieffer. 2014. A universal grammar-based code for lossless compression of binary trees. *IEEE Transactions on Information Theory* 60, 3 (March 2014), 1373–1386. DOI : <https://doi.org/10.1109/TIT.2013.2295392>

Received September 2017; revised July 2018; accepted August 2018