Lossless Compression of Binary Trees with Correlated Vertex Names

Abram Magner
Coordinated Science Laboratory
Univ. of Illinois at Urbana-Champaign
Urbana, IL, USA
Email: anmagner@illinois.edu

Krzysztof Turowski
Dept. of Algorithms and System Modeling
Gdansk University of Technology
Gdansk, Poland
Email: krzturow@pg.gda.pl

Wojciech Szpankowski
Department of Computer Science
Purdue University
West Lafayette, IN, USA
Email: spa@cs.purdue.edu

***Abstract*—Compression schemes for advanced data structures have become the challenge of today. Information theory has traditionally dealt with conventional data such as text, image, or video. In contrast, most data available today is multitype and context-dependent. To meet this challenge, we have recently initiated a systematic study of advanced data structures such as unlabeled graphs [1]. In this paper, we continue this program by considering trees with statistically correlated vertex names. Trees come in many forms, but here we deal with binary plane trees (where order of subtrees matters) and their non-plane version. Furthermore, we assume that each symbol of a vertex name depends in a Markovian sense on the corresponding symbol of the parent vertex name. We first evaluate the entropy for both types of trees. Then we propose two compression schemes** CompressPTree **for plane trees with correlated names, and** CompressNPTree **for non-plane trees. We prove that the former scheme achieves the lower bound within two bits, while the latter is within 1% of the optimal compression.**

I. Introduction

Over the last decade, repositories of various data have grown enormously. Most of the available data is no longer in conventional form, such as text or images. Instead, biological data (e.g., gene expression data, protein interaction networks, phylogenetic trees), medical data (cerebral scans, mammogram, etc.), and social network data archives are in the form of *multimodal* data structures, that is, multitype and context dependent structures. For efficient compression of such data structures, one must take into account not only several different types of information, but also the statistical dependence between the general data labels and the structures themselves.

This paper is a first step in the direction of the larger goal of a unified framework for compression of *labeled* graph and *tree* structures. We focus on compression of trees with structure-correlated vertex "names" (strings). First, we formalize natural models for random binary trees with correlated names. We focus on two variations: plane-oriented and non-plane-oriented trees. For the plane-oriented case, we give an exact formula for the entropy of the labeled tree, as well as an efficient compression algorithm whose expected code length is within two bits away from the optimal length. In the non-plane case, due to the typically large number of symmetries, the compression problem becomes more difficult, and in this conference version of the paper we focus on the case without vertex names. We prove upper and lower bounds on the entropy of such trees and provide an efficient compression algorithm whose expected code length is no more than 1% away from the entropy.

Regarding prior work, literature on tree and graph compression is quite scarce. For unlabeled graphs there are some recent information-theoretic results, including [2], [1] (see also [4]) and [3]. In 1990, Naor [2] proposed a representation for unlabeled graphs (solving Turan's [5] open question) that is asymptotically optimal when all unlabeled graphs are equally likely. Naor's result is asymptotically a special case of recent work of Choi and Szpankowski [1]. In [1] the entropy and optimal compression algorithm for Erdős-Rényi graphs were presented. Furthermore, in [3], an automata approach was used to design an optimal graph compression algorithm. There also have been some heuristic methods for real-world graph compression. Peshkin [6] proposed an algorithm for a graphical extension of the one-dimensional SEQUITUR compression method, however, it is known not to be asymptotically optimal. For binary plane-oriented trees rigorous information-theoretic results were obtained in [7], complemented by a universal grammar based lossless coding scheme [8].

However, for *labeled* structures, which is the topic of this paper, there have been almost no attempts at theoretical analyses, with the notable exception of [9] for sparse Erdős-Rényi graphs. The only relevant results have been based on heuristics, exploiting the well-known properties of special kinds of graphs [13]. Similarly, there are some algorithms with good practical compression rate for phylogenetic trees [14]; however, they too lack any theoretical guarantees on their performance.

II. Models

We call a rooted tree a *plane tree* when we distinguish left-to-right-order of the successors of the nodes in the embedding of a tree on a plane (see [12]). To avoid confusion, we call a tree with no fixed ordering of its subtrees a *non-plane* tree.

Let $\mathcal{T}$ be the set of all binary rooted plane trees having finitely many vertices and, for each positive integer n, let $\mathcal{T}_n$ be the subset of $\mathcal{T}$ consisting of all trees with exactly n leaves. Similarly, let $\mathcal{S}$ and $\mathcal{S}_n$ be the set of all binary rooted non-plane trees with finitely many vertices and exactly n leaves, respectively.

We can also augment our trees with vertex names – given the alphabet $\mathcal{A}$, names are simply words from $\mathcal{A}^m$ for some integer $m \geq 1$. Let $\mathcal{LT}_n$ and $\mathcal{LS}_n$ be the set of all binary rooted plane and non-plane trees with names, respectively, having exactly n leaves with each vertex assigned a name – a word from $\mathcal{A}^m$. In this paper we consider the case where names are locally correlated as discussed below.

Formally, a rooted plane tree $t \in \mathcal{T}_n$ can be uniquely described by its sequence of vertices $(v_i)_{i=1}^{2n-1}$ in depth-first search (DFS) order, together with a set of triples (v_i, v_j, v_k), consisting of the parent vertex and its left and right children. Furthermore, a tree with names $lt \in \mathcal{LT}_n$ can be uniquely described as a pair (t, f), where t is a tree, described as above, and f is a function from the vertices of t to the words in $\mathcal{A}^m$.

We now present a model for generating plane trees with locally correlated names. We shall assume that individual names are generated by a memoryless source (horizontal independence), but the letters of vertex names of children depend on the vertex name of the parent (vertical Markovian dependence).

Our main model MT_1 is defined as follows: given the number n of leaves in the tree, the length of the names m, the alphabet $\mathcal{A}$ (of size $|\mathcal{A}|$) and the transition probability matrix P (of size $|\mathcal{A}| \times |\mathcal{A}|$) with its stationary distribution π (i.e. $\pi P = \pi$), we define a random variable LT_n as a result of the following process: starting from a single node with a randomly generated name of length m by a memoryless source with distribution π, we repeat the following steps until a tree has exactly n leaves:

- pick uniformly at random a leaf v in the tree generated in the previous step,
- append two children v^L and v^R to v,
- generate correlated names for v^L and v^R by taking each letter from v and generating new letters according to P: for every letter of the parent we pick the corresponding row of matrix P and generate randomly the respective letters for v^L and v^R.

Our second model, MT_2 (also known as the binary search tree model), is ubiquitous in the computer science literature, arising for example in the context of binary search trees formed by inserting a random permutation of $[n-1]$ into a binary search tree. Under this model we generate a random tree T_n as follows: t is equal to the unique tree in $\mathcal{T}_1$ and we associate a number n with its single vertex. Then, in each recursive step, let $v_1, v_2, \ldots, v_k$ be the leaves of t, and let integers $n_1, n_2, \ldots, n_k$ be the values assigned to these leaves, respectively. For each leaf v_i with value $n_i > 1$, randomly select integer s_i from the set $\{1, \ldots, n_i - 1\}$ with probability $\frac{1}{n_i - 1}$ (independently of all other such leaves), and then grow two edges from v_i with the left edge terminating at a leaf of the extended tree with value s_i and the right edge terminating at a leaf of the extended tree with value $n_i - s_i$. The extended tree is the result of the current recursive step. Clearly, the recursion terminates with a binary tree having exactly n leaves, in which each leaf has assigned value 1; this tree is T_n. The assignment of names to such trees proceeds exactly as in the MT_1 model.

Let $\Delta(t)$ be the number of leaves of a tree t. For simplicity, let us also use $\Delta(v)$ to denote the number of leaves of a tree rooted at v. In [7] it was shown that under the model MT_2, $\mathbb{P}(T_n = t_1) = 1$ (where t_1 is the unique tree in $\mathcal{T}_1$) and

$$\mathbb{P}(T_n = t) = \frac{1}{n-1}\mathbb{P}(T_{\Delta(t^L)} = t^L)\mathbb{P}(T_{\Delta(t^R)} = t^R),$$

which leads us to the formula $\mathbb{P}(T_n = t) = \prod_{v \in \mathring{V}} (\Delta(v) - 1)^{-1}$ where $\mathring{V}$ is the set of the internal vertices of t. We can prove that both models are equivalent.

Theorem 1. *Under the model MT_1 it holds that $\mathbb{P}(T_n = t) = \prod_{v \in \mathring{V}} (\Delta(v) - 1)^{-1}$ where $\mathring{V}$ is the set of the internal vertices of t. Therefore, models MT_1 and MT_2 are equivalent.*

On the top of these models, we define models MS_1 and MS_2 for non-plane trees: just generate the tree according to MT_1 (respectively, MT_2) and then treat the resulting plane tree T_n as a non-plane one S_n.

III. Entropy evaluations

A. The entropy of the plane trees with names

Let $\Delta(LT_n^L)$ be a random variable corresponding to the number of leaves in the left subtree of LT_n. From the previous section, we know that $\mathbb{P}(\Delta(LT_n^L) = i) = \frac{1}{n-1}$. Let also r, r^L, r^R denote the root of LT_n and the roots of its left and right subtrees (denoted by LT_n^L and LT_n^R), respectively.

First, we observe that there is a bijection between a tree with names LT_n and a quadruple $(\Delta(LT_n^L), F_n(r), LT_n^L, LT_n^R)$ (that is, for each named tree, there is a unique associated quadruple, and each *valid* quadruple defines a unique tree). Therefore $H(LT_n|F_n(r)) = H(\Delta(LT_n^L), F_n(r), LT_n^L, LT_n^R|F_n(r))$. Second, using the fact that $\Delta(LT_n^L)$ is independent from

$F_n(r)$ and that LT_n^L and LT_n^R are independent, we find

$$H(LT_n|F_n(r)) = H(\Delta(LT_n^L)) + \sum_{k=1}^{n-1} \big(H(LT_n^L|F_n(r), \Delta(LT_n^L) = k) + H(LT_n^R|F_n(r), \Delta(LT_n^L) = k)\big)\, \mathbb{P}(\Delta(LT_n^L) = k). \quad (1)$$

By the definition, we know that $H(\Delta(LT_n^L)) = \log_2(n-1)$.

Next, it is sufficient to note that the random variable LT_n^L conditioned on $\Delta(LT_n^L) = k$ is identical to the random variable LT_k (the model has the hereditary property), and the same holds for LT_n^R and LT_{n-k}. Similarly, F_n on the left (or right) subtree of LT, given $F_n(r^L)$ (respectively, $F_n(r^R)$) and $\Delta(LT_n^L) = k$ has identical distribution to the random variable F_k (resp. F_{n-k}) given $F_k(r)$ ($F_{n-k}(r)$). This leads us to the following recurrence:

$$H(LT_n|F_n(r)) = \log_2(n-1) + 2mh(P) + \frac{2}{n-1}\sum_{k=1}^{n-1} H(LT_k|F_k(r)) \quad (2)$$

where $h(P) = -\sum_{a\in\mathcal{A}} \pi(a) \sum_{b\in\mathcal{A}} P(b|a)\log P(b|a)$ is the entropy of the Markov process with the transition matrix P.

In (2) we encounter a recurrence whose general solution is presented next.

Lemma 1. *The recurrence* $x_1 = a_1$,

$$x_n = a_n + \frac{2}{n-1}\sum_{k=1}^{n-1} x_k, \quad n \geq 2 \quad (3)$$

has the following solution:

$$x_n = a_n + n\sum_{k=1}^{n-1} \frac{2a_k}{k(k+1)}. \quad (4)$$

Proof: By comparing the equations for x_n and x_{n+1} for any $n \geq 2$ we obtain

$$\frac{x_{n+1}}{n+1} = \frac{x_n}{n} + \frac{na_{n+1} - (n-1)a_n}{n(n+1)}.$$

Substituting $y_n = \frac{x_n}{n}$ and $b_n = \frac{na_{n+1}-(n-1)a_n}{n(n+1)}$ we find

$$y_n = y_1 + \sum_{k=1}^{n-1} b_k = y_1 + \sum_{k=1}^{n-1}\left(\frac{a_{k+1}}{k+1} - \frac{a_k}{k} + \frac{2a_k}{k(k+1)}\right)$$
$$= y_1 + \frac{a_n}{n} - a_1 + \sum_{k=1}^{n-1}\frac{2a_k}{k(k+1)} = \frac{a_n}{n} + \sum_{k=1}^{n-1}\frac{2a_k}{k(k+1)}$$

which completes the proof. ∎

Using Lemma 1 we immediately obtain our first main result.

Theorem 2. *The entropy of a binary tree with names, generated according to the model* MT_1*, is given by*

$$H(LT_n) = \log_2(n-1) + 2n\sum_{k=2}^{n-1}\frac{\log_2(k-1)}{k(k+1)} + 2(n-1)mh(P) + mh(\pi) \quad (5)$$

where $h(\pi) = -\sum_{a\in\mathcal{A}} \pi(a)\log\pi(a)$.

We know that $2\sum_{k=2}^{\infty}\frac{\log_2(k-1)}{k(k+1)} \approx 1.736$. Therefore, if we consider trees without names ($m = 0$), the entropy of the trees would be approximately equal $1.736n$, as was already shown in [7]. In a typical setting $m = O(1)$ or $\Theta(\log n)$ (which is certainly needed if we want all the labels to be distinct) – so $H(LT_n)$ would be $O(n)$ or $O(n\log n)$, respectively.

B. The entropy of the non-plane trees

Now we turn our attention to the non-plane trees and the case when $m = 0$. Let S_n be the random variable with probability distribution given by the MS_1 (or, equivalently, MS_2) model. For any $s \in \mathcal{S}$ and $t \in \mathcal{T}$ let $t \sim s$ mean that the plane tree t is isomorphic to the non-plane tree s. Furthermore, we define $[s] = \{t \in \mathcal{T} : t \sim s\}$. For any $t_1, t_2 \in \mathcal{T}_n$ such that $t_1 \sim s$ and $t_2 \sim s$ it holds that $\mathbb{P}(T_n = t_1) = \mathbb{P}(T_n = t_2)$ [11]. By definition, s corresponds to $|[s]|$ isomorphic plane trees, so for any $t \sim s$ it holds that $\mathbb{P}(S_n = s) = |[s]|\mathbb{P}(T_n = t)$.

Let us now introduce two functions: $X(t)$ and $Y(t)$, which are equal to the number of internal vertices of $t \in \mathcal{T}$ with unbalanced subtrees (unbalanced meaning that the numbers of leaves of its left and right subtree are not equal) and the number of internal vertices with balanced, but non-isomorphic subtrees, respectively. Similarly, let $X(s)$ and $Y(s)$ denote the number of such vertices for $s \in \mathcal{S}$. Clearly, $X(t) = X(s)$ and $Y(t) = Y(s)$. Moreover, observe that any structure $s \in \mathcal{S}$ has exactly $|[s]| = 2^{X(s)+Y(s)}$ distinct plane orderings, since each vertex with non-isomorphic subtrees can have either one of two different orderings of the subtrees, whereas when both subtrees are isomorphic – only one.

Now we can conclude that:

$$H(T_n|S_n) = -\sum_{t\in\mathcal{T}_n, s\in\mathcal{S}_n} \mathbb{P}(T_n = t)\log\mathbb{P}(T_n = t|S_n = s) = \sum_{t\in\mathcal{T}_n}\mathbb{P}(T_n = t)(X(t) + Y(t)) = \mathbb{E}X_n + \mathbb{E}Y_n$$

where we write $X_n := X(T_n)$ and $Y_n := Y(T_n)$.

C. Lower and upper bounds on $H(T_n|S_n)$

Lower Bound. The stochastic recurrence for X_n can be expressed as follows: $X_1 = 0$ and

$$X_n = X_{U_{n-1}} + X_{n-U_{n-1}} + I\left(U_{n-1} \neq \frac{n}{2}\right)$$

for $n \geq 2$, where U_n is the variable on $\{1, 2, \ldots, n\}$ with uniform distribution. This leads us to

$$\mathbb{E}X_n = \frac{1}{n-1}\sum_{k=1}^{n-1}\mathbb{E}(X_n|U_{n-1} = k)$$
$$= \mathbb{E}I\left(U_{n-1} \neq \frac{n}{2}\right) + \frac{2}{n-1}\sum_{i=1}^{n-1}\mathbb{E}X_i.$$

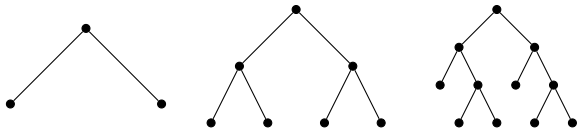


Fig. 1: An example of $\mathfrak{s}_1 * \mathfrak{s}_1$, $\mathfrak{s}_2 * \mathfrak{s}_2$, $\mathfrak{s}_3 * \mathfrak{s}_3$

Since, $\mathbb{E}I\left(U_{n-1} \neq \frac{n}{2}\right) = (n-2)/(n-1)$ for n even (and 1 for n odd except for $n = 1$ when it is zero), we find by Lemma 1 setting $x_n = \mathbb{E}X_n$ and $a_n = \mathbb{E}I\left(U_{n-1} \neq \frac{n}{2}\right)$

$$\mathbb{E}X_n = a_n + n\left(2 - \frac{2}{n} - \sum_{k=1}^{n-1} \frac{2(-1)^{k+1}}{k}\right) \sim 2n(1 - \ln 2).$$

Upper Bound. Now we turn our attention to the upper bound on $H(T_n|S_n)$. We can introduce another function $Z(t)$ – the number of internal vertices of t with isomorphic subtrees. Obviously, $X(t) + Y(t) + Z(t) = n - 1$. Given $\mathfrak{s} \in \mathcal{S}$ we may define $Z(t, \mathfrak{s})$ – the number of internal vertices with both subtrees isomorphic to $\mathfrak{s}$. Clearly, $Z(t) = \sum_{\mathfrak{s} \in \mathcal{S}} Z(t, \mathfrak{s})$, and $Z(T_n) = \sum_{\mathfrak{s} \in \mathcal{S}} Z(T_n, \mathfrak{s})$. We may bound the value of $\mathbb{E}Z_n := \mathbb{E}Z(T_n)$ from below by computing only particular values of $\mathbb{E}Z_n(\mathfrak{s}) := \mathbb{E}Z(T_n, \mathfrak{s})$

Let us also use $\mathfrak{s} * \mathfrak{s}$ as a shorthand for a non-plane tree having both subtrees of a root isomorphic to $\mathfrak{s}$. The stochastic recurrence on $Z_n(\mathfrak{s})$ becomes $Z_n(\mathfrak{s}) = I\left(T_n \sim \mathfrak{s} * \mathfrak{s}\right) + Z_{U_{n-1}}(\mathfrak{s}) + Z_{n-U_{n-1}}(\mathfrak{s})$ which leads us to

$$\mathbb{E}Z_n(\mathfrak{s}) = \mathbb{E}I\left(T_n \sim \mathfrak{s} * \mathfrak{s}\right) + \frac{2}{n-1} \sum_{k=1}^{n-1} \mathbb{E}Z_k(\mathfrak{s}).$$

Moreover, since under condition that $\Delta(T_n^L) = k$ the event that $T_n \sim \mathfrak{s} * \mathfrak{s}$ is equivalent to the union of the events $T_n^L \sim \mathfrak{s}$ and $T_n^R \sim \mathfrak{s}$, we have

$$\begin{aligned}\mathbb{E}I\left(T_n \sim \mathfrak{s} * \mathfrak{s}\right) &= \frac{1}{n-1} \sum_{k=1}^{n-1} \mathbb{P}\left(T_n \sim \mathfrak{s} * \mathfrak{s} | U_{n-1} = k\right) \\ &= I\left(n = 2\Delta(\mathfrak{s})\right) \frac{\mathbb{P}^2(T_{n/2} \sim \mathfrak{s})}{n-1}.\end{aligned}$$

Now, we apply Lemma 1 with $x_n = \mathbb{E}Z_n(\mathfrak{s})$ and $a_n = I\left(n = 2\Delta(\mathfrak{s})\right) \frac{\mathbb{P}^2(T_{n/2} \sim \mathfrak{s})}{n-1}$. Its solution (for $n \geq 2\Delta(\mathfrak{s})$) is

$$\mathbb{E}Z_n(\mathfrak{s}) = \frac{2\mathbb{P}^2(T_{\Delta(\mathfrak{s})} \sim \mathfrak{s})n}{(2\Delta(\mathfrak{s}) - 1)2\Delta(\mathfrak{s})(2\Delta(\mathfrak{s}) + 1)}. \tag{7}$$

If we consider only three subtrees: $\mathfrak{s}_1$ (cherries), $\mathfrak{s}_2$ and $\mathfrak{s}_3$ (presented on Fig. 1), it can be easily noticed that $\mathbb{P}(T_1 \sim \mathfrak{s}_1) = \mathbb{P}(T_2 \sim \mathfrak{s}_2) = \mathbb{P}(T_3 \sim \mathfrak{s}_3) = 1$. Therefore for any $n \geq 6$ it holds that $\mathbb{E}Z_n(\mathfrak{s}_1) = \frac{n}{3}$ (see also [11] for a proof), $\mathbb{E}Z_n(\mathfrak{s}_2) = \frac{n}{30}$, $\mathbb{E}Z_n(\mathfrak{s}_3) = \frac{n}{105}$ so $\mathbb{E}Z_n \geq \mathbb{E}Z_n(s_1) + \mathbb{E}Z_n(\mathfrak{s}_2) + \mathbb{E}Z_n(\mathfrak{s}_3) = \frac{79}{210}n$ – so $\mathbb{E}X_n + \mathbb{E}Y_n = n - 1 - \mathbb{E}Z_n \leq 0.6239n$. In summary we obtain the following result.

Theorem 3. *The conditional entropy of a plane tree from MT_1 given its structure is asymptotically bounded as follows:*

$$0.6137 \leq \lim_{n\to\infty} \frac{H(T_n|S_n)}{n} \leq 0.6239.$$

Therefore, the entropy of a non-plane tree generated under model MS_1 is asymptotically bounded as follows:

$$1.112 \leq \lim_{n\to\infty} \frac{H(S_n)}{n} \leq 1.123.$$

Because $H(T_n) - H(S_n) = H(T_n|S_n)$, on average the compression of the structure (the non-plane tree) requires asymptotically $0.6137n$ fewer bits than the compression of any plane tree isomorphic to it.

IV. Algorithms

The main idea of the two algorithms presented here for plane and non-plane trees is to use the arithmetic coding scheme.

A. Algorithm CompressPTree for plane trees with names

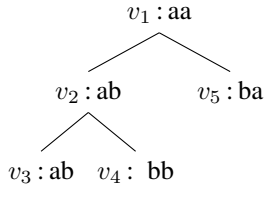
First we set the interval to $[0, 1)$ and then traverse the vertices of the tree and its names. Here, we explicitly assume that we traverse the tree in DFS order, first descending to the left subtree and then to the right.

At each step, if we visit an internal node v, we split the interval according to the probabilities of the number of leaves in the left subtree. That is, if the subtree rooted at v has k leaves and v^L has l leaves, then we split the interval into $k-1$ equal parts and pick l-th subinterval as the new interval. Then if v is the root, for every letter of the name $F_n(v)$ we split the interval according to the distribution π and pick the subinterval representing the respective letter of $F_n(v)$. If v is not the root, for every letter of the name $F_n(v)$, we split the interval according to transition probability distribution in P according to the respective letter in $F_n(u)$, where u is a parent of v, and pick the subinterval representing the letter of $F_n(v)$. Finally, when we obtain an interval $[a, b)$, we take as output of the algorithm the first $\lceil -\log_2(b-a) \rceil$ bits of $\frac{a+b}{2}$.

Example. Suppose that $\mathcal{A} = \{a, b\}$ with the transition matrix

$$P = \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}. \tag{8}$$

Observe that $\pi = (0.4, 0.6)$. Then for the following tree our algorithm CompressPTree proceeds as follows:



v_1	$\Rightarrow [0.5, 1)$
a	$\Rightarrow [0.5, 0.7)$
a	$\Rightarrow [0.5, 0.58)$
v_2	$\Rightarrow [0.5, 0.58)$
a $\to$ a	$\Rightarrow [0.5, 0.556)$
...	

and finally, we obtain $[0.550282624, 0.5507567872)$ – so we take the middle point of it (0.5505197056) and return its first $\lceil -\log(0.5507567872 - 0.550282624) \rceil + 1 = 13$ bits in the binary representation, that is 1000110011101.

B. Algorithm CompressNPTree for non-plane trees

For non-plane trees, we present a suboptimal compression algorithm called CompressNPTree that for a non-plane tree S_n *without* vertex names, produces a code from which we can efficiently reconstruct a non-plane tree.

As before, we first set the interval to $[0, 1)$ and then traverse the vertices of the tree. However, now we always visit the smaller subtree (that is, the one with the smaller number of leaves) first (ties are broken arbitrarily). At each step, if we visit an internal node v, we split the interval according to the probabilities of the sizes of the smaller subtree – that is, if the subtree rooted at v has k leaves and its smaller subtree has l leaves, then if k is even we split the interval into $\frac{k}{2}$ equal parts. Otherwise, k is odd, so we split the interval into $\frac{k+1}{2}$ parts, all equal, except the last one, which has half the size of the others. Finally, in both cases we pick the l-th subinterval as the new interval.

C. Analysis of the algorithm for plane trees

In the journal version of our paper we prove the following.

Theorem 4. *For a given plane tree with names lt the algorithm* CompressPTree *computes an interval whose length is equal to $\mathbb{P}(LT_n = lt)$.*

Therefore, if we take the first $\lceil -\log \mathbb{P}(LT_n = lt) \rceil + 1 = \lceil -\log(b-a) \rceil + 1$ bits of the middle of the interval $[a, b)$ we get a codeword $C_n^{(1)}$, which is guaranteed to be inside this interval and which is uniquely decodable.

Theorem 5. *The average length of a codeword $C_n^{(1)}$ of* CompressPTree *is only within 2 bits from the entropy.*

Proof: From the analysis above, we know that:

$$\mathbb{E}C_n^{(1)} = \sum_{lt \in \mathcal{LT}_n} \mathbb{P}(LT_n = lt)(\lceil -\log_2 \mathbb{P}(LT_n = lt) \rceil + 1) <$$

$$\sum_{lt \in \mathcal{LT}_n} -\mathbb{P}(LT_n = lt) \log_2 \mathbb{P}(LT_n = lt) + 2 = H(LT_n) + 2$$

which completes the proof. ∎

D. Analysis of the algorithm for non-plane trees

We now deal with CompressNPTree algorithm for the non-plane trees together with the analysis of its performance. The algorithm does not match the entropy rate for the non-plane trees, since for every vertex with two non-isomorphic subtrees of equal sizes, it can visit either subtree first – resulting in different output intervals, so different codewords too. Clearly, such intervals are shorter than the length of the optimal interval; therefore, the codeword will be longer. However, given the bounds on $\mathbb{E}Y_n$, we can bound the expected redundancy rate within 1% of the entropy rate.

If we want to obtain an optimal algorithm for the compression of non-plane trees, we need a unique way of resolving the ties for vertices which have non-isomorphic subtrees with the same number of leaves. Unfortunately, if we fix any linear ordering on the trees of the same size and force to go to the smaller one first, it skews the probability distribution for the right subtree, as it reveals some information about it. We can prove the following.

Lemma 3. *For a given non-plane tree s with exactly $Y(s)$ vertices with balanced but not isomorphic subtrees,* CompressNPTree *computes an interval whose length is equal to $2^{-Y(s)}\mathbb{P}(S_n = s)$.*

As before, if we use the arithmetic coding scheme on an interval generated from a tree s, we get a codeword $C_n^{(2)}$, which, as a binary number, is guaranteed to be inside this interval. Moreover, we can prove the following.

Theorem 6. *The average length of a codeword $C_n^{(2)}$ from algorithm* CompressNPTree *does not exceed $1.01H(S_n)$.*

Acknowledgment

This work was supported by NSF Center for Science of Information (CSoI) Grant CCF-0939370, by NSF Grant CCF-1524312, and NIH Grant 1U01CA198941-01. W. Szpankowski is also with the Faculty of ETI, Gdańsk University of Technology, Poland.

References

[1] Y. Choi and W. Szpankowski: Compression of Graphical Structures: Fundamental Limits, Algorithms, and Experiments. *IEEE Transactions on Information Theory*, 2012, 58(2):620-638.

[2] M. Naor, Succinct representation of general unlabeled graphs, *Discrete Applied Mathematics*, 28(3), 303–307, 1990.

[3] M. Mohri, M. Riley, A. T. Suresh, Automata and graph compression. *ISIT 2015*, pp. 2989-2993.

[4] B. Guler, A. Yener, P. Basu, C. Andersen, A. Swami, A study on compressing graphical structures. *GlobalSIP 2014*, pp. 823-827

[5] Gy. Turan, On the succinct representation of graphs, *Discrete Applied Mathematics*, 8(3), 289–294, 1984.

[6] L. Peshkin, Structure induction by lossless graph compression, *In Proc. of the IEEE Data Compression Conference*, 53–62, 2007.

[7] J. C. Kieffer, E.-H. Yang, W. Szpankowski, Structural complexity of random binary trees. *ISIT 2009* , pp. 635-639.

[8] J. Zhang, E.-H. Yang, J. C. Kieffer, A Universal Grammar-Based Code for Lossless Compression of Binary Trees. *IEEE Transactions on Information Theory*, 2014, 60(3):1373-1386.

[9] D. Aldous and N. Ross, Entropy of Some Models of Sparse Random Graphs With Vertex-Names. *Probability in the Engineering and Informational Sciences*, 2014, 28:145-168.

[10] M. Steel, A. McKenzie, Distributions of cherries for two models of trees. *Mathematical Biosciences*, 2000, 164:81-92.

[11] M. Steel, A. McKenzie, Properties of phylogenetic trees generated by Yule-type speciation models. *Mathematical Biosciences*, 2001, 170(1):91-112.

[12] M. Drmota, *Random Trees: An Interplay between Combinatorics and Probability*. Springer Publishing Company, Inc., 2009.

[13] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan. On Compressing Social Networks, *Proc. ACM KDD*, 2009.

[14] S. J. Matthews, S.-J. Sul, T. L. Williams, TreeZip: A New Algorithm for Compressing Large Collections of Evolutionary Trees, *Data Compression Conference* 2010, pp. 544-554.