Combinatorial and Information-Theoretic Aspects of Tree Compression

DISSERTATION
zur Erlangung des Grades eines Doktors
der Naturwissenschaften

vorgelegt von

Louisa Kathrin Seelbach Benkner, M.Sc.

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2023

Betreuer und erster Gutachter
Prof. Dr. Markus Lohrey
Universität Siegen

Zweiter Gutachter
Prof. Dr. Stephan Wagner
Uppsala University, Schweden

Tag der mündlichen Prüfung
19. Oktober 2023

Abstract

We analyze lossless tree compression algorithms under information-theoretic and combinatorial aspects. One of the most important and widely used compression methods for rooted trees is to represent a tree by its minimal directed acyclic graph, shortly referred to as minimal DAG. The size of the minimal DAG of the tree is the number of distinct fringe subtrees occurring in the tree, where a fringe subtree of a rooted tree is a subtree induced by one of the nodes and all its descendants. In the first part of this work, we study the average number of distinct fringe subtrees (i.e., the average size of the minimal DAG) in random trees. Specifically, we consider the random tree model of leaf-centric binary tree sources as introduced by Kieffer et al. and prove upper and lower bounds for the average number of distinct fringe subtrees that hold for large classes of leaf-centric binary tree sources. Furthermore, we consider the random tree models of simply generated trees and families of increasing trees (recursive trees, d-ary increasing trees and generalized plane-oriented recursive trees), and prove that the order of magnitude of the average number of distinct fringe subtrees (under rather mild assumptions on what "distinct" means) in random trees with n nodes is $n/\sqrt{\log n}$ for simply generated trees and $n/\log n$ for increasing trees.

Among others, our results generalize a result by Flajolet et al. and Devroye, according to which the average number of distinct fringe subtrees in a random binary search tree of size n is in $\Theta(n/\log n)$, as well as a result by Flajolet et al., according to which the average number of distinct fringe subtrees in a uniformly random binary tree of size n is in $\Theta(n/\sqrt{\log n})$, in several ways.

In the second part of this work, we analyze grammar-based tree compression via tree straight-line programs (TSLPs) from an information-theoretic point of view. Specifically, we extend the notion of k^{th}-order empirical entropy from strings to node-labeled binary and plane trees and show that a suitable binary encoding of TSLPs yields binary tree encodings of size bounded by the k^{th}-order empirical entropy plus some lower order terms. This generalizes recent results from grammar-based string compression to grammar-based tree compression. Additionally, we carry out a systematic comparison of several different notions of empirical entropy for trees that have been proposed in the past, both from a theoretical and an experimental point of view.

In the third part of this work, we present a new compressed encoding of unlabeled binary and plane trees. We analyze this encoding under an information-theoretic point of view by proving that this encoding is universal and thus asymptotically optimal for a great variety of tree sources; this covers in particular the vast majority of tree sources with respect to which previous tree encodings were shown to be universal. At the same time, and in contrast to previous universal tree codes, our new compressed tree encoding can be turned into a tree data structure that supports answering many navigational queries on the compressed representation in constant time on the word-RAM. This solves in particular an open problem from Davoodi et al. in the context of optimal data structures for range-minimum queries.

Zusammenfassung

Wir analysieren verlustfreie Methoden der Baumkomprimierung unter informationstheoretischen und kombinatorischen Gesichtspunkten. Eine weit verbreitete Methode der Baumkomprimierung ist die sogenannte DAG-Komprimierung, bei der ein Baum durch seinen zugehörigen minimalen gerichteten azyklischen Graphen (engl. directed acyclic graph, kurz DAG) dargestellt wird. Die Größe dieses minimalen DAGs eines Baums ist die Anzahl der verschiedenen fringe subtrees des Baums. Ein fringe subtree eines gewurzelten Baums ist ein Teilbaum, der von einem der Knoten inklusive aller seiner Nachkommen induziert wird.

Im ersten Teil der Arbeit untersuchen wir die erwartete Anzahl der verschiedenen fringe subtrees (d.h., die durchschnittliche Größe des minimalen DAGs) bzgl. verschiedener Wahrscheinlichkeitsverteilungen auf verschiedenen Baumfamilien. Wir betrachten das Modell der leaf-centric tree sources, das auf Kieffer et al. zurückgeht, und beweisen obere und untere Schranken für die erwartete Anzahl verschiedener fringe subtrees, die für große Klassen von leaf-centric tree sources gelten. Weiterhin betrachten wir das Modell der simply generated trees und drei spezifische Modelle der increasing trees (recursive trees, d-ary increasing trees und generalized plane-oriented recursive trees). Wir zeigen, dass die erwartete Anzahl der verschiedenen fringe subtrees (unter einer verallgemeinerten Interpretation von "verschieden") in random simply generated trees asymptotisch wie $\Theta(n/\sqrt{\log n})$ wächst, und in random increasing trees asymptotisch wie $\Theta(n/\log n)$ wächst. Wir verallgemeinern mit unseren Ergebnissen ein Resultat von Flajolet et al. und Devroye, welches aussagt, dass die erwartete Anzahl von verschiedenen fringe subtrees in einem zufälligen binären Suchbaum der Größe n in $\Theta(n/\log n)$ ist, und ein Resultat von Flajolet et al., nach dem die erwartete Anzahl an verschiedenen fringe subtrees in einem bzgl. der Gleichverteilung zufällig gewählten Binärbaum der Größe n in $\Theta(n/\sqrt{\log n})$ ist.

Im zweiten Teil der Arbeit analysieren wir grammatik-basierte Baumkompression durch sogenannte tree straight-line programs (TSLPs). Wir erweitern den Begriff der empirischen Entropie von Wörtern auf Bäume und zeigen, dass eine geeignete Binärkodierung von TSLPs binäre Baumkodierungen liefert, deren Größe in der empirischen Entropie (plus lower-order terms) beschränkt ist. Dies verallgemeinert ein Resultat aus dem Gebiet der grammatikbasierten Wortkompression. Zusätzlich führen wir einen Vergleich verschiedener Notationen der empirischen Baumentropie durch, die in der Literatur betrachtet wurden.

Im dritten Teil der Arbeit stellen wir eine neue komprimierte Darstellung von Bäumen vor, die universal und daher optimal bezüglich einer großen Anzahl an Baumverteilungen ist; insbesondere gilt dies auch für die Mehrzahl der Verteilungen, bezüglich derer für bisherige Baumkodierungen Universalität nachgewiesen werden konnte. Im Gegensatz zu bisherigen universalen Baumkodierungen kann unsere neue komprimierte Baumdarstellung zusätzlich zu einer Datenstruktur erweitert werden, die Navigationsoperationen in konstanter Zeit im word-RAM Modell ausführt. Dies löst insbesondere ein offenes Problem von Davoodi et al., im Kontext optimaler Datenstrukturen für Range-Minimum Queries.

Acknowledgements

First of all, I would like to thank my advisor, Markus Lohrey. I am deeply grateful for his excellent guidance, for the freedom he gave me in choosing my research topics while at the same time always providing help and support, and for everything I have learned from him. I could always rely on his invaluable advice, irrespective of the subject matter, whenever I needed it.

I am deeply grateful to Stephan Wagner, who co-examined this thesis and shared with me his mathematical insights into random trees. Moreover, he gave me the wonderful opportunity to visit him for a research stay at the University of Stellenbosch in South Africa. Likewise, I am very thankful to Travis Gagie, Ian Munro and Gonzalo Navarro, for hosting me on amazing research stays at the University of Chile and at the University of Waterloo in Canada, and to Dieter Spreen for making these research stays possible.

Many thanks go to my coauthor Sebastian Wild, for countless interesting discussions, both research related and non-research related, and for inviting me to his workshop at the University of Liverpool. Also, I would like to thank my coauthor Jarno Niklas Alanko, for the fun time we had together while working on Wheeler graphs.

I am very grateful to my current and former colleagues at the University of Siegen, Michael Figelius, Moses Ganardi, Danny Hucke, Seungbum Jo, Julio Caesar Juarez Xochitemol, Daniel König, Carl Philipp Reh, Andreas Rosowski, Kurt Sieber and Tobias Schüler, who created a wonderful work atmosphere with many enjoyable conversations and interesting discussions.

Last but not least, I owe a lot of thanks to my friends, for their lasting friendship, to Robin, for the wonderful years we spend together, and to my family, for their loving care, endless encouragement and constant support.

Contents

Chapter 1

Introduction

As the amount of digital data around the world increases rapidly, storing data efficiently has become a fundamental challenge in computer science. A classic research area in this context is the field of *lossless data compression*, where the goal is to devise algorithms that efficiently compress data with no loss of information, such that the original data can be perfectly retrieved from its compressed representation.

In this work, we focus on lossless compression of *rooted trees* as one of the most fundamental types of structured data in computer science. In many cases, data features a hierarchical structure, which can be naturally represented as a tree; a classic example in this context are the document trees corresponding to XML documents. Moreover, various data structures and algorithms involve trees; well-known examples include binary search trees, red-black trees and AVL trees, suffix trees, Wavelet trees, and Cartesian trees, see, e.g., [67] and [84] for details.

Two of the most widely used lossless compression algorithms for trees are *DAG compression* and *grammar-based tree compression*. In this work, we investigate the compression performance of these tree compressors from a combinatorial, respectively, information-theoretic point of view. Furthermore, we propose and analyze a new tree compression technique based on the tree covering algorithm from [30]. The results of this work can be roughly divided into three main topics.

1.1 DAG compression and fringe subtrees

A *fringe subtree* of a rooted tree is a subtree which consists of a node and all its descendants. Fringe subtrees are a natural object of study in the context of random trees, and there is a great variety of results for various random tree models, see e.g. [4, 23, 25, 31].

Fringe subtrees are of particular interest in computer science. A widely used lossless compression method for rooted trees is to represent a tree as a directed acyclic graph (DAG), which is obtained by merging nodes that are roots of identical fringe subtrees (see Figure 1.1 for an example). This compressed representation of the tree is often shortly referred to as *minimal DAG* of the tree

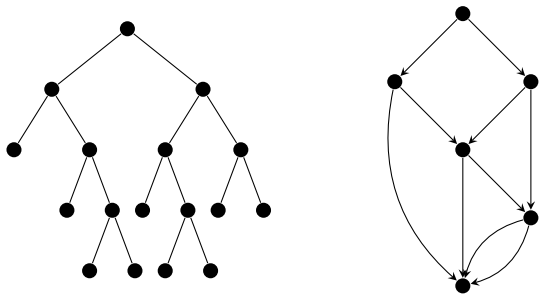


Figure 1.1: A binary tree (left) and its corresponding minimal DAG (right).

and its size (number of nodes) is the number of distinct fringe subtrees occurring in the tree. Compression by minimal DAGs has found applications in many areas of computer science, as for example in compiler construction [3, Chapter 6.1 and 8.5], unification [93], symbolic model checking (binary decision diagrams) [13], information theory [S3], [105], and XML compression and querying [14, 39].

The minimal DAG of a tree is unique and can be computed in linear time [26]. The size of a minimal DAG (when measured as its number of nodes) can be exponentially smaller than the size of the corresponding tree (consider, for example, a perfect binary tree). However, the size of a minimal DAG of a tree can also be linear in the size of its corresponding tree: take, for example, a linear chain tree. Thus, it seems natural to investigate the *average size* (i.e., average number of nodes) of the minimal DAG, or, equivalently, the average number of distinct fringe subtrees in random trees. So far, this problem has mainly been investigated with respect to the following random tree models.

In [38], Flajolet, Sipala and Steyaert proved that, under very general assumptions, the expected number of distinct fringe subtrees in a tree of size n drawn *uniformly at random* from some given family of trees is asymptotically equal to $c \cdot n/\sqrt{\log n}$, where the constant c depends on the particular family of trees. In particular, their result covers uniformly random plane trees (where the constant c evaluates to $c = \sqrt{2/\pi}$) and uniformly random binary trees (with $c = 2\sqrt{2/\pi}$). The result of Flajolet et al. was extended to uniformly random Σ-labeled unranked trees in [12] (where Σ-labeled means that each node of a tree is assigned a label from a finite alphabet Σ and unranked means that the label of a node does not depend on its degree or vice versa). Moreover, an alternative proof to the result of Flajolet et al. was presented in [96] in the context of simply generated families of trees.

Another probabilistic tree model with respect to which the number of distinct fringe subtrees has been studied is the *binary search tree model*. A random binary search tree of size n is a binary search tree built by inserting the keys $\{1, \ldots, n\}$ according to a uniformly chosen random permutation on $\{1, \ldots, n\}$. Random binary search trees are of particular interest in computer science, as they naturally arise for example in the analysis of the Quicksort algorithm, see [27]. In [36], Flajolet, Gourdon and Martinez proved that the expected number of distinct fringe subtrees in a random binary search tree of size n is $O(n/\log n)$.

This result was improved in [24] by Devroye, who showed that the asymptotics $\Theta(n/\log n)$ holds. In a recent paper [9], the result of Flajolet, Gourdon and Martinez was reproved. Moreover, it is shown in [9] for *random recursive trees* (defined in Chapter 5), that the average number of distinct fringe subtrees in a random tree of size n is also in $\mathcal{O}(n/\log n)$ and bounded from below by $\Omega(\sqrt{n})$ and it is conjectured that the asymptotics $\Theta(n/\log n)$ holds.

In this thesis, we generalize the results from [9, 24, 36, 38, 96] in several ways. In Chapter 3, we focus on random full binary trees, i.e., rooted ordered trees, such that each node has either exactly two or zero descendants, and derive upper and lower bounds on the average number of distinct fringe subtrees that hold for large classes of distributions. More precisely, we consider the random tree model of *leaf-centric binary tree sources*, which was introduced in [66] (see also [105]), as a very general concept to model probability distributions on the set of full binary trees with n leaves[1]. Both the binary search tree model as well as the uniform probability distribution on the set of full binary trees with n leaves can be modeled as a leaf-centric binary tree source [66].

In particular, we consider four classes of leaf-centric binary tree sources and derive asymptotic upper or lower bounds on the average number of distinct fringe subtrees occurring in a random tree generated by a leaf-centric binary tree source from the respective class. These upper or lower bounds thus cover not only a single distribution, but a whole class of distributions. An overview over the four classes of leaf-centric binary tree sources and the respective results is given in Section 3.1.

These results generalize the results from [24, 36] on the binary search tree model in the sense that they imply the upper bound $\mathcal{O}(n/\log n)$ (respectively, the lower bound $\Omega(n/\log n)$) on the expected number of distinct fringe subtrees in a random binary search tree of size n. Also, from two of the results combined we obtain that the expected number of distinct fringe subtrees in a uniformly random full binary tree with n leaves is in $\Theta(n/\sqrt{\log n})$, as shown in [38, 96].

In Chapter 4 we then consider the random tree model of random simply generated trees [27, 58, 82] as a general concept to model uniform probability distributions on various families of trees and in Chapter 5, we focus on specific families of increasing trees (recursive trees, d-ary increasing trees and generalized plane oriented recursive trees), which in particular incorporate the binary search tree model [7, 27].

In all of the results [9, 24, 36, 38, 96] mentioned above, the problem of deriving asymptotic estimates for the expected number of distinct fringe subtrees in random trees is studied with respect to the *specific notion of distinctness* that two fringe subtrees are considered as distinct if they are distinct as members of the particular family of trees. In Chapter 4 and Chapter 5, we estimate the number of "distinct" fringe subtrees in random trees under a generalized interpretation of "distinctness", which allows for many different interpretations of what "distinct" trees are.

[1] As every full binary tree with n leaves has exactly $2n-1$ nodes in total, it is often convenient to indicate its size by its number of leaves.

The main motivation for this generalized notion of distinctness is based on a problem in the context of XML compression: Here, one distinguishes between document-centric XML, for which the corresponding XML document trees are ordered (i.e., an ordering is specified for the children of each node), and data-centric XML, for which the corresponding XML document trees are unordered (i.e., there is no ordering on the nodes' children). Understanding the interplay between ordered and unordered structures has thus received considerable attention in the context of XML (see for example [1, 11, 106]). In particular, in [75], it was investigated whether tree compression can benefit from unorderedness. For this reason, *unordered minimal DAGs* were considered. An unordered minimal DAG of a tree is a directed acyclic graph obtained by merging nodes that are roots of fringe subtrees which are identical as unordered trees. From such an unordered minimal DAG, an unordered representation of the original tree can be uniquely retrieved. The size of this compressed representation is the number of distinct unordered trees represented by the fringe subtrees in the tree. So far, only some worst-case estimates comparing the size of a minimal DAG to the size of its corresponding unordered minimal DAG are known. Among other things, it was shown in [75] that the size of an unordered minimal DAG of a binary tree can be exponentially smaller than the size of the corresponding (ordered) minimal DAG.

However, no average-case estimates comparing the size of the minimal DAG of a tree to the size of the corresponding unordered minimal DAG are known so far. In particular, in [75] it is stated as an open problem to estimate the expected number of distinct unordered trees represented by the fringe subtrees of a uniformly random binary tree of size n and conjectured that this number asymptotically grows as $\Theta(n/\sqrt{\log n})$.

As one of our main theorems of Chapter 4 (following from a more general main theorem), we settle this open conjecture by proving upper and lower bounds of order $n/\sqrt{\log n}$ for the number of distinct unordered trees represented by the fringe subtrees of a tree of size n drawn randomly from a simply generated family of trees, which hold both in expectation and with high probability. Furthermore, we show that the results from [38, 96] on the number of distinct fringe subtrees (as members of the particular family) in simply generated trees do not only hold in expectation, but also with high probability.

Similarly, as one of our main theorems of Chapter 5 (also following from a general main theorem), we prove upper and lower bounds of order $n/\log n$ for the number of distinct unordered trees represented by the fringe subtrees of a random binary search tree of size n. Additionally, we improve the results from [36, 24] on the number of distinct (as binary trees) fringe subtrees in random binary search trees, by showing that these results do not only hold in expectation, but also with high probability, and by providing an improved lower bound (in terms of the leading constant). Finally, also in Chapter 5, we solve the open problem from [9], by proving that the number of distinct fringe subtrees in a random recursive tree of size n is $\Theta(n/\log n)$ in expectation and with high probability.

1.2 Grammar-based tree compression and empirical tree entropy

Tree compression by minimal DAGs has the disadvantageous shortcoming that it only allows to exploit repeated occurrences of *fringe subtrees* in the tree, however, repeated patterns "inside" the tree are not taken advantage of. A tree compression method that allows to avoid this shortcoming is *grammar-based tree compression*, which represents a natural generalization of DAG compression.

The idea of grammar-based compression is based on the fact that in many cases a string s can be succinctly represented by a context-free grammar that produces only the string s. Such a grammar is called a straight-line program (SLP) for s. In the best case, the size of an SLP for a string of length n can be in $\Theta(\log n)$, where the size of an SLP is measured as the total length of all right-hand sides of the rules of the grammar. A grammar-based compressor is an algorithm that produces for a given string s an SLP $\mathcal{G}_s$, where $\mathcal{G}_s$ should be smaller than s. The first systematic investigations of grammar-based compressors are carried out in [17, 64]. Grammar-based string compressors can be found at many places in the literature, probably the best known example is the LZ78-compressor of Lempel and Ziv [107]. Other well-known grammar-based compressors are Bisection [63], Sequitur [87], and Repair [72], to mention a few.

Grammar-based compression has been generalized from strings to trees by means of linear context-free tree grammars generating exactly one tree [15]. Such grammars are also known as tree straight-line programs, TSLPs for short, see [73] for a survey. Tree straight-line programs do not only represent a generalization of (string) SLPs, but also generalize DAGs in the sense that TSLPs allow to share repeated tree patterns occurring "inside" the tree. Any DAG for a tree t can be seen as a TSLP for t, but with the restriction that every nonterminal produces a fringe subtree of t. For general TSLPs, this restriction is loosened, as nonterminals are also allowed to produce *contexts*, which are fringe subtrees of t which may contain one (or several) "holes". The precise formalism of tree compression by TSLPs will be introduced in Chapter 6.

TSLPs allow exponential compression in the best case, and due to the ability to share also internal patterns, there are examples where the minimal DAG is exponentially larger than the smallest TSLP of a tree [73]. In [42], the authors present a linear time algorithm that computes for a given ranked (constant maximal degree) tree t of size n a TSLP $\mathcal{G}_t$ of size $\mathcal{O}(n/\log_{\hat{\sigma}} n)$, where σ is the size of the underlying set of node labels and $\hat{\sigma} = \max\{2, \sigma\}$. An alternative algorithm with the same asymptotic size bound can be found in [44].

TSLPs have been extended to *forest straight-line programs* (FSLPs), which allow to decompose trees "horizontally" (splitting the children of one node) and hence allow to compress node-labeled plane trees (of unbounded degree). They meet the worst-case size bound $\mathcal{O}(n/\log_{\hat{\sigma}} n)$ for node-labeled plane trees of size n [47]. Moreover, we remark that grammar-based tree compression is closely related to the tree compression formalism of *top DAGS*, see e.g. [8, 47, 76].

Note that the worst-case bound $\mathcal{O}(n/\log_{\hat{\sigma}} n)$ cannot be achieved by DAG

compression: the smallest DAG for an unlabeled full binary tree of leafsize n may still contain n nodes. Moreover, a simple information-theoretic argument shows that the worst-case bound of the form $\mathcal{O}(n/\log_{\hat{\sigma}} n)$ (for labeled plane or ranked trees of size n) achieved by TSLPs, respectively, FSLPs, cannot be improved to a smaller asymptotic worst-case bound. However, a more refined worst-case analysis of grammar-based tree compression is possible using the concept of *empirical entropy*.

In the area of string compression, the concept of higher order empirical entropy yields a well-established measure for the compressibility of a string. Empirical entropy for strings was introduced by Kosaraju and Manzini in [70]. Intuitively, the k^{th}-order empirical entropy of a string s is our expected uncertainty about the symbol at a certain position in the string, given the k preceding symbols, see e.g. [41]. "Empirical" refers to the fact that the entropy is defined for the string itself and not for a certain probability distribution on strings. This has the advantage that empirical entropy is also useful in situations where we do not know the underlying probability distribution on strings.

Empirical entropy for strings is a well-established tool in order to analyze the worst-case compression performance of string compressors (for further aspects of empirical entropy for strings, see [41]). For many string compressors, worst-case bounds on the length of a compressed string s in terms of the k^{th}-order empirical entropy (plus lower-order terms) are known, as for example for the Burrows-Wheeler transform [80], several algorithms from the LZ-family [70], and also for grammar-based string compressors [45, 86, 89]. These kinds of upper bounds are often referred to as "entropy bounds". Intuitively, if a string compressor satisfies an entropy bound, the string compressor tends to exploit the kind of regularities in strings that the k^{th}-order empirical entropy is able to capture.

In Chapter 6 of this work, we prove an entropy bound for grammar-based tree compressors. For this, a reasonable notion of empirical entropy for trees is needed. In recent years, several attempts were made to generalize the concept of empirical entropy from strings to trees; specifically, there is the concept of *label entropy* from [32], the *degree entropy* from [60], and their combinations *label-degree entropy* and *degree-label entropy* from [46]. However, all of these notions of empirical tree entropy have some limitations and only incorporate partial aspects of trees, as they are tailored towards plane trees. In particular, label entropy is not suitable for unlabeled trees, degree entropy only incorporates the structure of the tree, but not the node labels, and all of the entropy notions from [32, 60, 46] do not work for important special cases as unlabeled full binary trees.

For these reasons, one of our main contributions of Chapter 6 is the definition of a new reasonable entropy measure for labeled trees that can be also used for the unlabeled case. Our notion of empirical tree entropy incorporates both node labels as well as the shape of the tree and is in particular able to capture dependencies between node labels and tree shape, hence, we call it *label-shape entropy*. As our main result of Chapter 6, we show that the encoding length of a particular type of grammar-based tree compressor can be bounded from

above in terms of the k^{th}-order label-shape entropy plus some lower order terms. In particular, this generalizes a recent result [89] from grammar-based string compression to grammar-based tree compression.

In Chapter 7 we then carry out a systematic comparison, both from a theoretical as well as from an experimental point of view, between the entropy notions from [32, 60, 46] and our new notion of label-shape entropy. In particular, we show that label-shape entropy lower bounds the existing entropy notions from [32, 60, 46] for the special case of labeled and unlabeled full binary trees, and that for the special case of unlabeled plane trees, label-shape entropy is never greater than twice the entropy notion from [60] plus lower-order terms. Moreover, we carry out experiments with real XML data, which indicate that an upper bound on the number of bits needed by a compressed tree representation in terms of the label-shape entropy is the strongest for real XML data, since the k^{th} order label-shape entropy (for $k > 0$) is significantly smaller than all other entropy bounds for all XML documents that we have examined.

1.3 Tree data structures and universal tree source coding

For many applications, it is desirable to not only compress the input data, but also to be able to efficiently answer queries on the compressed representation without prior decompression. For this reason, various compressed tree representations have been turned into *data structures*. Tree data structures based on grammar-based compression and top DAGs typically achieve logarithmic query times in the word-RAM model for the navigational queries they support.

On the other hand, a great variety of *succinct tree data structures* has been proposed in the literature, notably BP (balanced parenthesis) [56, 83], DFUDS [6, 60], LOUDS [56], tree covering [48, 52, 30] and their variants and enhancements (see also [22, 84, 98] for an overview), which usually achieve constant query times on the word-RAM. However, a typical property of succinct data structures is that their space usage is determined only by the size of the input, that is, they use $\log a_n(1 + o(1))$ bits of space in order to represent one out of a_n objects of size n, such that there is no further compression beyond the worst-case entropy. For example, standard succinct tree data structures use $2n + o(n)$ bits of space for any tree with n nodes. Notable exceptions are the tree data structures from [60, 46], which support navigational queries in constant time on the word-RAM and at the same time achieve an entropy bound in terms of the degree entropy [60], respectively, in terms of the entropy notions from [46].

In Chapter 8 and Chapter 9, we propose a new compressed encoding for unlabeled binary and plane trees, called *hypersuccinct trees*, which can be turned into a data structure that answers a large number of navigational queries in constant time on the word-RAM. It is based on the tree decomposition algorithm from [30]. Hypersuccint trees achieve entropy bounds in terms of the label-shape entropy (introduced in Chapter 6 of this work), as well as in terms of the degree

entropy from [60] (which is identical to the entropy notions from [46] for the special case of unlabeled trees). What sets the hypersuccinct tree data structure apart from previous data structures [60, 46], is that it can be shown to be *universal* with respect to a great variety of tree sources.

Universal source coding for finite sequences over a finite alphabet Σ is a well-established topic in information theory. The central object of study in (classical) information theory is that of a source of random strings. A fundamental result in information theory states that the minimum possible average code length for binary prefix-free lossless codes is the *Shannon entropy* [20] corresponding to the respective source. In the same way, the minimum possible worst-case code length is the *self-information* [20] of the encoded objects with respect to the respective source. A significant goal in an information-theoretic sense in compressing such strings is a *universal code*, which achieves optimal compression (up to lower order terms) for distributions of strings from a large class of possible sources *without knowing the used source.*

In a series of papers, grammar-based string codes that are universal for the class of finite state sources were developed [63, 64, 65]. Similar results hold for Lempel-Ziv methods [107] and the Burrows-Wheeler-transform [28]. Over the last few years, we have seen increasing efforts aiming to extend universal source coding to structured data like trees [66, 105, 79, 49] and graphs [18, 77]. In particular, universal source coding for unlabeled binary trees based on DAG-compression and grammar-based tree compression has been considered in [105] and [S3] (a detailed overview over their results is given in Section 8.1). However, compared to the situation for strings, universal source coding and information theory of structured data is much less developed.

In Chapter 8 and Chapter 9 of this work, we show that the hypersuccinct tree encodings are universal with respect to a great variety of tree sources. In the case of binary trees, this includes in particular classes of *leaf-centric binary tree sources*, which we also consider in Chapter 3 of this work in the context of DAG compression. Specifically, hypersuccinct binary trees can be shown to be universal with respect to the vast majority of tree sources, for which previous universal tree encodings for unlabeled binary trees have been shown to be universal [105], [S3]. In contrast to hypersuccinct binary trees, these previous universal binary tree codes cannot be turned into a data structure with constant query times for navigational operations, as recent lower bounds [95] imply. In the case of plane trees, no universal tree source codes specifically focusing on plane trees have been proposed before.

Hence, the hypersuccinct tree encodings proposed in Chapters 8 and Chapter 9 of this work are the first compressed representation of binary, respectively plane trees that can be shown to be universal with respect to a large number of classes of tree sources and at the same time can be turned into a data structure that supports a wide range of navigational queries in constant time on the word-RAM.

Chapter 2

Preliminaries

2.1 Basic notation

With $\mathbb{N}$ we denote the natural numbers without zero, and with $\mathbb{N}_0$ we denote the natural numbers including zero. The logarithm of a positive number x to the base b is denoted by $\log_b x$, and the natural logarithm of a positive number x is denoted by $\ln x$. Moreover, the binary logarithm of a positive number x is denoted by $\log x$. We use the standard Landau notations $\mathcal{O}$, o, Ω, ω and Θ.

An alphabet Σ is a nonempty set of *characters* (also called *symbols*), which is usually finite. A *word* or *string* over an alphabet Σ is a finite sequence $w = a_1 a_2 \cdots a_l$ of (not necessarily distinct) characters $a_1, a_2, \ldots, a_l$. With $|w| = l$ we denote the length of w. We write Σ^* for the set of all words over Σ, with ϵ we denote the empty word, and we write Σ^n for the set of all words of length n over Σ. If Σ is finite, we denote the size of Σ with σ. For $a \in \Sigma$ we denote with $|w|_a = |\{i \mid 1 \leq i \leq l, a_i = a\}|$ the number of occurrences of a in w.

2.2 Trees

A tree is a connected acyclic undirected graph. A tree is called *rooted*, if one specific node of the tree is marked as the root node. This induces an ancestor-descendant relationship on the nodes of the tree, such that each node (except for the root node) has a uniquely defined parent node. The direct descendants of a node are called its *children*. Throughout this work, all trees we consider will be rooted. In particular, if not explicitly stated otherwise, the term "tree" will always denote a rooted tree.

A rooted tree is called *ordered*, if there is an ordering on the children of each node of the tree. Likewise, if there is no such ordering, a tree is called *unordered*. Most trees we consider in this work will be ordered. If a tree is unordered, it will be mentioned explicitly.

The *size* $|t|$ of a tree t is defined as its number of nodes. Sometimes it is convenient to consider the *empty tree*, which is a tree of size zero. Let v be a node of a tree t, then we denote with $\deg(v)$ the *degree* of the node v, that is,

the number of children of the node v. A node that does not have any children is called a *leaf* and a node that has at least one child node is called an *inner node* or *internal node*. With $V(t)$ we denote the set of nodes of a tree t and with $V_0(t)$ we denote the set of inner nodes of t.

The *depth* of a node v of a tree t is the length of the path from the root node to v. The *depth* $d(t)$ of a tree t is the maximal depth of a node v of t. We define the *height* $\hbar(t)$ of a tree t to be the depth $d(t)$ plus one (thus, the height of a tree of size one is one). Moreover, we define the height of the empty tree to be zero.

A *subtree* of a tree t is a connected acyclic subgraph of t. A *fringe subtree* of a tree t is a subtree of t that consists of a node of t together with all its descendants. For a node v of t, we denote with $t[v]$ the fringe subtree of t that is rooted in v.

2.2.1 Plane trees and binary trees

There is a great variety of different families of trees. The most important families of trees that we will consider in this work are the following:

Definition 2.1 (Plane tree)**.** A *plane tree* is a rooted, ordered tree. With $\mathcal{T}$ we denote the set of all plane trees and with $\mathcal{T}_n$ we denote the set of all plane trees of size n for $n \geq 1$.

Plane trees are sometimes called *ordinal trees*, see e.g. [6]. A plane tree is shown in Figure 2.1 on the left.

Definition 2.2 (Full binary tree)**.** A *full binary tree* is a rooted, ordered tree, such that each node has either exactly two or zero children. With $\mathcal{B}$ we denote the set of all full binary trees and with $\mathcal{B}_n$ we denote the set of all full binary trees with n leaves for $n \geq 1$.

A full binary tree is shown in Figure 2.1 (second tree from the left). As every node in a full binary tree has either exactly two or zero children, we find that every full binary tree with $n-1$ inner nodes has exactly n leaves. The total size of a full binary tree is thus always an odd number. It is hence often convenient to measure the size of a full binary tree as its number of leaves (thus, $\mathcal{B}_n$ denotes the set of all full binary trees with n leaves for $n \geq 1$). With $\|t\|$ we denote the number of leaves of a tree t. The number of leaves of a tree t is often called its *leafsize*. Moreover, if t is a full binary tree, we denote with $t_l[v]$, respectively, $t_r[v]$ the fringe subtree rooted in the left, respectively, right child node of an inner node v of t. If v is the root node, we write t_l and t_r for $t_l[v]$ and $t_r[v]$. We shortly refer to t_l and t_r as the *left subtree*, respectively, *right subtree* of t.

We remark again that full binary trees always denote ordered trees. Unordered trees such that each node has exactly two or zero descendants will be called *unordered full binary trees*, that is, it will be stated explicitly that the considered trees are unordered.

Definition 2.3 (Binary tree)**.** A *binary tree* is a rooted, ordered tree, such that each node has either exactly two children, a single left child, a single right child, or zero children. With $\mathcal{B}^{\diamond}$ we denote the set of all binary trees and with $\mathcal{B}_n^{\diamond}$ we denote the set of all binary trees of size n for $n \geq 1$.

Note that we distinguish between left-unary and right-unary nodes in binary trees. A binary tree is shown in Figure 2.1 (the third tree from the left). We transfer the notions $t_l[v]$, $t_r[v]$ and t_l, t_r from full binary trees to binary trees (however, in this case, these fringe subtrees can be the empty tree).

Furthermore, note that whereas $\mathcal{B}_n$ denotes the set of full binary trees with n *leaves*, $\mathcal{B}_n^{\diamond}$ denotes the set of binary trees of *size* n, that is, with n nodes in total. There is a natural one-to-one correspondence between the set $\mathcal{B}_n$ of full binary trees with n leaves and the set $\mathcal{B}_{n-1}^{\diamond}$ of binary trees of size $n-1$ for every $n \geq 1$. The corresponding bijection maps a full binary tree t of leafsize n to the binary tree of size $n-1$ that is obtained by removing all of t's leaves and only keeping the inner nodes of t. For example, the full binary tree in Figure 2.1 (the second tree from the left) corresponds to the binary tree in Figure 2.1 (the third tree from the left). In particular, in view of this one-to-one correspondence, many results shown for the family of full binary trees also become applicable for the family of binary trees and vice versa. In this thesis, we will consider both full binary trees and (not necessarily full) binary trees, as for some results and applications, it will be more convenient or more natural to consider one or the other.

Binary trees, full binary trees and plane trees are enumerated by the *Catalan numbers*:

Definition 2.4 (Catalan numbers)**.** The *Catalan numbers* are a sequence $(\mathscr{C}_n)_{n\geq 0}$ of natural numbers defined by

$$\mathscr{C}_n = \frac{1}{n+1}\binom{2n}{n}.$$

The Catalan numbers (see e.g. [37]) satisfy the following recursion

$$\mathscr{C}_{n+1} = \sum_{i=0}^{n} \mathscr{C}_i \mathscr{C}_{n-i}$$

for $n \geq 0$. Moreover, asymptotically, the Catalan numbers grow as

$$\mathscr{C}_n \sim \frac{4^n}{\sqrt{\pi} n^{3/2}} \left(1 + \mathcal{O}(1/n)\right). \tag{2.1}$$

It is a well known fact in combinatorics on trees (see e.g. [27, 37]) that

$$|\mathcal{B}_n| = |\mathcal{B}_{n-1}^{\diamond}| = |\mathcal{T}_n| = \mathscr{C}_{n-1}.$$

Binary trees and full binary trees are generalized to d-ary trees and full d-ary trees:

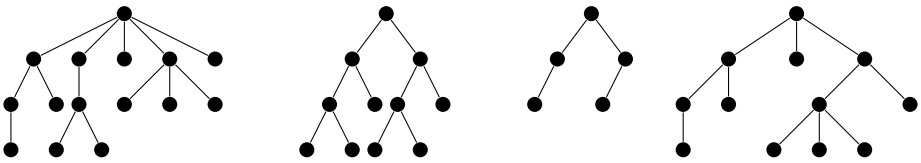


Figure 2.1: From left to right: A plane tree, a full binary tree, a binary tree and a 3-ary tree.

Definition 2.5 (d-ary trees and full d-ary trees)**.** Let $d \geq 2$ be an integer. A *d-ary tree* is a rooted ordered tree, such that each node has exactly d possible positions to which children can be attached. Likewise, a *full d-ary tree* is a rooted ordered tree such that each node has either exactly d children or zero children.

Thus, the degree of a node v of a d-ary tree is bounded from above by d and there are $\binom{d}{k}$ types of nodes of degree k for $0 \leq k \leq d$. For example, if $d = 3$, a node of degree $k = 2$ can have a left and a middle child, a left and a right child, or a right and a middle child. A 3-ary tree is shown in Figure 2.1 on the right.

Sometimes d-ary trees are called *cardinal trees of degree d* [6], in contrast to ordinal trees, to emphasize that each node has d positions to which nodes can be attached, whereas for ordinal trees (i.e., plane trees), we do not distinguish different positions to which child nodes can be attached.

2.2.2 Labeled trees and formal expressions

The term "labeled tree" is not always used consistently in the literature. In this thesis, a labeled tree denotes the following concept.

Definition 2.6 (Σ-labeled tree)**.** Let Σ be an alphabet. A Σ-labeled tree is a tree, where every node is assigned a label from the alphabet Σ.

In general, Σ-labeled trees can be ordered or unordered, however, we will mostly consider ordered Σ-labeled trees in this work. If Σ is clear from the context or arbitrary, we will often simply use the term "labeled tree" instead of "Σ-labeled tree". The *label* of a node v of a tree will be denoted with $\lambda(v)$. Important families of Σ-labeled trees are the following.

Definition 2.7 (Σ-labeled full binary trees)**.** A *Σ-labeled full binary tree* is a full binary tree such that each node is assigned a label from the alphabet Σ. With $\mathcal{B}(\Sigma)$ we denote the set of Σ-labeled full binary trees and with $\mathcal{B}_n(\Sigma)$ we denote the set of Σ-labeled full binary trees with n leaves.

A Σ-labeled full binary tree is shown in Figure 2.2 on the left. In the same way, we define the set of Σ-labeled plane trees:

Definition 2.8 (Σ-labeled plane trees)**.** A *Σ-labeled plane tree* is a plane tree such that each node is assigned a label from the alphabet Σ. With $\mathcal{T}(\Sigma)$ we denote the set of Σ-labeled plane trees and with $\mathcal{T}_n(\Sigma)$ we denote the set of Σ-labeled plane trees of size n.

Figure 2.2: A Σ-labeled full binary tree with $\Sigma = \{a, b, c\}$ (left) and a numbered tree (right).

In particular, the label of a node of a tree $t \in \mathcal{B}(\Sigma)$ or $t \in \mathcal{T}(\Sigma)$ does not determine its degree or vice-versa. To stress the fact that the label of a node does not define its degree, Σ-labeled plane trees are sometimes referred to as *unranked trees* (see, e.g., [12, 47]). In contrast, *ranked trees* are Σ-labeled trees, such that each character a from the alphabet Σ is assigned a number $\mathrm{rank}(a) \in \mathbb{N}_0$ (called the *rank* of a), and such that a node that is labeled with the character a always has $\mathrm{rank}(a)$ children. If not explicitly stated otherwise, the labeled trees we consider in this work will not be ranked.

In the context of labeled trees, an unlabeled tree can be considered as a Σ-labeled tree over a unary alphabet, that is, we identify the sets $\mathcal{B}$, $\mathcal{B}^\diamond$ and $\mathcal{T}$ with $\mathcal{B}(\{a\})$, $\mathcal{B}^\diamond(\{a\})$ and $\mathcal{T}(\{a\})$ for some character a. The *shape* of a labeled tree t is the underlying unlabeled tree, that is, the tree obtained from t by removing all node labels.

Sometimes it is convenient to identify Σ-labeled trees with formal expressions over the alphabet Σ. When considered as formal expressions, the set $\mathcal{B}(\Sigma)$ of Σ-labeled full binary trees is defined as follows.

Definition 2.9 (Σ-labeled full binary trees as formal expressions)**.** The set $\mathcal{B}(\Sigma)$ of Σ-labeled full binary trees over the alphabet Σ is inductively defined as the smallest set of terms over Σ such that

- $\Sigma \subseteq \mathcal{B}(\Sigma)$ and
- if $t_1, t_2 \in \mathcal{B}(\Sigma)$ and $a \in \Sigma$, then $a(t_1, t_2) \in \mathcal{B}(\Sigma)$.

For example, if $\Sigma = \{a, b\}$, then the term a corresponds to the tree which consists of only one single node which is labeled with a. The tree in Figure 2.2 on the left corresponds to the term $c(b(b(a, c), b(a, a)), a(c, a))$.

Finally, we will consider a specific family of Σ-labeled unordered trees over the infinite alphabet $\Sigma = \mathbb{N}$:

Definition 2.10 (Numbered tree)**.** An *numbered* tree of size n is an unordered rooted tree, such that each node is assigned a label from the set $\{1, \ldots, n\}$ and no two nodes are labeled with the same label.

We stress again that numbered trees are always unordered by definition. A (plane representation of a) numbered tree is shown in Figure 2.2 on the left. Numbered trees are often called labeled trees (see, e.g., [27, 97]), however, for the sake of clarity, we use the term "numbered tree" here.

2.2.3 Forests and the first-child next-sibling encoding

For technical reasons, it will sometimes be convenient to consider *forests*.

Definition 2.11 (Forests)**.** A *forest* is a (possibly empty) sequence of plane trees. The set of all forests is denoted with $\mathcal{F}$.

The *size* $|f|$ of a forest f is defined as the sum of the sizes of the trees of the forest and with $\mathcal{F}_n$ we denote the set of all forests of size n. A Σ-labeled forest is a forest such that each node obtains a label from the alphabet Σ. With $\mathcal{F}(\Sigma)$ (respectively, $\mathcal{F}_n(\Sigma)$), we denote the set of all Σ-labeled forests (respectively, Σ-labeled forests of size n). When considered as formal expressions, the sets $\mathcal{F}(\Sigma)$ and $\mathcal{T}(\Sigma)$ are defined as follows:

Definition 2.12 (Σ-labeled plane trees and forests as formal expressions)**.** The sets $\mathcal{T}(\Sigma)$ and $\mathcal{F}(\Sigma)$ are inductively defined as the smallest sets of terms over Σ such that

- $\epsilon \in \mathcal{F}$ (this is the empty forest),
- if $a \in \Sigma$ and $f \in \mathcal{F}(\Sigma)$, then $a(f) \in \mathcal{T}(\Sigma)$,
- if $t \in \mathcal{T}(\Sigma)$ and $f \in \mathcal{F}(\Sigma)$, then $tf \in \mathcal{F}(\Sigma)$.

A bijective mapping between the set $\mathcal{F}_n(\Sigma)$ and the set $\mathcal{B}_{n+1}(\Sigma)$ is established via the *first-child next-sibling encoding*. For this, we fix a distinguished character $\square \in \Sigma$ (the choice will be arbitrary for our purposes).

Definition 2.13 (First-child next-sibling encoding)**.** The first-child next-sibling encoding transforms a forest $f \in \mathcal{F}(\Sigma)$ into a full binary tree $\mathrm{fcns}(f) \in \mathcal{B}(\Sigma)$. It is inductively defined as

- $\mathrm{fcns}(\epsilon) = \square$ (where ϵ again denotes the empty forest) and
- $\mathrm{fcns}(a(f)g) = a(\mathrm{fcns}(f), \mathrm{fcns}(g))$ for $f, g \in \mathcal{F}$ and $a \in \Sigma$.

Thus, the left (respectively, right) child of a node in $\mathrm{fcns}(f)$ is the first child (respectively, right sibling) of the node in f or a $\square$-labeled leaf if the node in f does not have a first child (respectively, right sibling). Note that the definition of the first-child next-sibling encoding is also reasonable for unlabeled trees (by identifying them with labeled trees over a unary alphabet as mentioned before).

2.2.4 Random trees

A *random tree* T is a random variable taking values in some set of trees according to a probability distribution on this set of trees. There is a great variety of random tree models (see, e.g., [7, 27, 58, 66, 82, 105]).

Very natural random tree models are those based on *uniform distributions*:

Definition 2.14 (Uniformly random plane/binary/full binary trees)**.** A *uniformly random plane/binary/full binary tree* T_n of (leaf-)size n is a random tree that takes values in the set $\mathcal{T}_n$, respectively, $\mathcal{B}_n^\diamond$, respectively $\mathcal{B}_n$ according to the uniform probability distribution on the respective set.

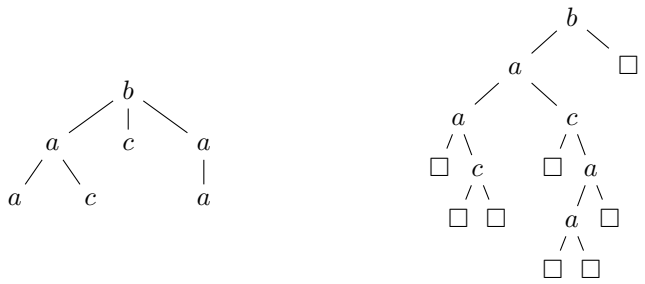


Figure 2.3: A Σ-labeled plane tree with $\Sigma = \{a, b, c\}$ (left) and its first-child next-sibling encoding (right), where $\square$ is an arbitrary character in Σ.

Another important and well-known random tree model is the following model, which is defined for binary trees and full binary trees.

Definition 2.15 (Binary search tree model)**.** The binary search tree model is the random tree model induced by the probability mass function $P_{bst} \colon \mathcal{B}_n^\diamond \to [0, 1]$ (for every $n \geq 1$) with

$$P_{bst}(t) = \prod_{v \in V(t)} \frac{1}{|t[v]|}.$$

The binary search tree model for full binary trees corresponds to the probability mass function $P_{bst} \colon \mathcal{B}_n \to [0, 1]$ (for every $n \geq 1$) defined by

$$P_{bst}(t) = \prod_{v \in V_0(t)} \frac{1}{\|t[v]\| - 1}.$$

The binary search tree model for full binary trees is obtained from the binary search tree model that generates (not necessarily full) binary trees via the one-to-one correspondence between the sets $\mathcal{B}_n$ and $\mathcal{B}_{n-1}^\diamond$ and vice-versa. Note that as the second product ranges over all inner nodes $v \in V_0(t)$, we find that the leafsize of the fringe subtrees satisfies $\|t[v]\| > 1$, such that the product is well-defined.

The random binary search tree model arises quite naturally in computer science (see, e.g., [27]): A random binary search tree of size n is a binary search tree built by inserting the keys $\{1, \ldots, n\}$ according to a uniformly chosen random permutation on $\{1, \ldots, n\}$.

In this work, we consider various random tree models that generalize the models from Definition 2.14 and Definition 2.15 in several ways. We also consider further random tree models that take node labels into account. Since most random tree models will only be considered in one or two chapters of this work, we defer further definitions of random tree models to the respective chapters.

Part I

Fringe Subtrees in Random Trees

Chapter 3

Leaf-centric binary tree sources

3.1 Introduction and preliminary definitions

In this chapter, we are interested in the expected number of *distinct fringe subtrees* in random full binary trees. In particular, in this chapter, we focus exclusively on full binary trees, that is, ordered rooted trees, such that each node has either exactly two or zero children (see Definition 2.2). However, via the one-to-one correspondence between the sets $\mathcal{B}$ and $\mathcal{B}^{\diamond}$ (see Section 2.2.1), all results and concepts presented in this chapter easily transfer to binary trees with left-unary and right-unary nodes (Definition 2.3) as well.

Formally, the number of distinct fringe subtrees occurring in a full binary tree $t \in \mathcal{B}$ can be defined as follows: We define an equivalence relation $\sim$ on the set of nodes of t by $v \sim v'$ if and only if the two fringe subtrees $t[v]$ and $t[v']$ are identical as ordered full binary trees. Let $[v]$ denote the equivalence class of node v with respect to this equivalence relation. Then the number of distinct fringe subtrees occurring in t equals the number of equivalence classes $|\{[v] \mid v \in V(t)\}|$. See Figure 3.1 for an example of a full binary tree and its distinct fringe subtrees.

So far, the average number of distinct fringe subtrees in random full binary trees has mainly been studied with respect to two random tree models: uniformly random full binary trees, and random binary search trees (see Definitions 2.14 and 2.15). In [38], Flajolet, Sipala and Steyaert proved that the expected number of distinct fringe subtrees in a uniformly random full binary tree of leafsize n is asymptotically equal to $c \cdot n/\sqrt{\log n}$, where c is the constant $c = 2\sqrt{2/\pi}$. An alternative proof to the result of Flajolet et al. was presented in [96] in the context of simply generated families of trees (which will be considered in Chapter 4 of this thesis).

For random binary search trees, Flajolet, Gourdon and Martinez proved in [36] that the expected number of distinct fringe subtrees in a random (full)

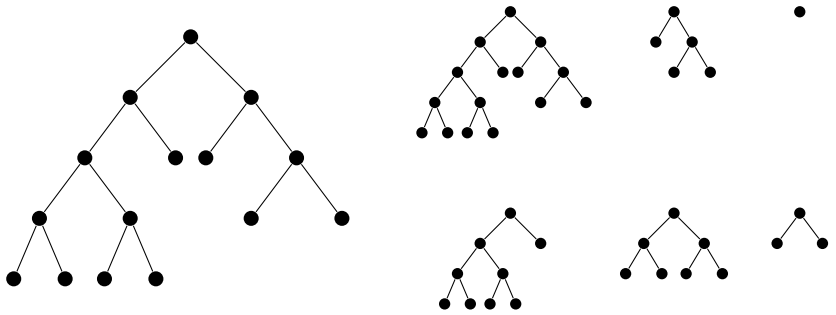


Figure 3.1: A full binary tree (left) and its six distinct fringe subtrees (right).

binary search tree of leafsize n is bounded from above by $(4n/\log n)(1+o(1))$. This result was improved in [24] by Devroye, who provided an alternative proof for the upper bound and showed that the asymptotics $\Theta(n/\log n)$ holds by deriving a lower bound of the form $(cn/\log n)(1+o(1))$ for the constant $c = \log(3)/2 \approx 0.7924812504$. Moreover, in a recent paper, an upper bound of the form $\mathcal{O}(n/\log n)$ was reproved by Bodini et al. [9].

The goal of this chapter is to generalize the results from [24, 36, 38] by proving upper and lower bounds on the average number of distinct fringe subtrees in random full binary trees that hold for large classes of distributions. A very general concept to model probability distributions on the set of full binary trees of leafsize n was presented by Kieffer et al. in [66] (see also [105]), where the authors introduce *leaf-centric binary tree sources* as an extension of the classical notion of an information source on finite sequences to full binary trees. Formally, a leaf-centric binary tree source is defined as follows.

Definition 3.1 (Leaf-centric binary tree source [105, 66])**.** Let $\mathcal{L}$ denote the set of all functions $\ell\colon \mathbb{N}\times\mathbb{N}\to[0,1]$ that satisfy

$$\sum_{\substack{i,j\geq 1\\ i+j=k}} \ell(i,j) = 1 \tag{3.1}$$

for every integer $k \geq 2$. A mapping $\ell \in \mathcal{L}$ induces a probability mass function $P_\ell\colon \mathcal{B}_n \to [0,1]$ for every $n \geq 1$ in the following way: define $P_\ell\colon \mathcal{B}\to[0,1]$ inductively by

$$P_\ell(t) = \begin{cases} 1 & \text{if } \|t\| = 1,\\ \ell(\|t_l\|,\|t_r\|)\cdot P_\ell(t_l)\cdot P_\ell(t_r) & \text{otherwise.}\end{cases} \tag{3.2}$$

A tuple $(\mathcal{B},(\mathcal{B}_n)_{n\in\mathbb{N}},P_\ell)$ with $\ell\in\mathcal{L}$ is called a *leaf-centric tree source*.

In other words, ℓ restricted to the set of ordered pairs $\{(i,n-i)\mid 1\leq i\leq n-1\}$ is a probability mass function for every $n\geq 2$. We sometimes refer to the mapping ℓ itself as a *leaf-centric binary tree source*. Intuitively, a leaf-centric binary tree source randomly generates a full binary tree of leafsize n as follows. We start at the root node and determine the leafsizes of the left and of the right subtree,

where the probability that the left subtree is of leafsize i for $i \in \{1, \ldots, n-1\}$ (and consequently, the right subtree is of leafsize $n-i$) is given by $\ell(i, n-i)$. This process then recursively continues in the left and right subtree, that is, the leaf-centric binary tree source then randomly generates a full binary tree of leafsize i as the left subtree and a full binary tree of leafsize $n-i$ as the right subtree. In particular, for $t \in \mathcal{B}$, we have

$$P_\ell(t) = \prod_{v \in V_0(t)} \ell(\|t_l[v]\|, \|t_r[v]\|).$$

Both the uniform distribution on $\mathcal{B}_n$ (Definition 2.14) and the random binary search tree model (Definition 2.15) can be modeled as leaf-centric binary tree sources:

Example 3.2. The *binary search tree model* corresponds to the leaf-centric binary tree source defined by

$$\ell_{bst}(i, n-i) = \frac{1}{n-1}$$

for every $n \geq 2$ and $1 \leq i \leq n-1$ (see [66]).

Example 3.3. Recall the definition of the Catalan numbers $(\mathscr{C}_n)_{n \geq 0}$ from Definition 2.4. The *uniform distribution* on the sets $\mathcal{B}_n$ corresponds to the leaf-centric binary tree source with

$$\ell_{uni}(i, n-i) = \frac{\mathscr{C}_{i-1}\mathscr{C}_{n-i-1}}{\mathscr{C}_{n-1}}$$

for every $n \geq 2$ and $1 \leq i \leq n-1$ [66]. In particular, we obtain

$$P_{\ell_{uni}}(t) = \frac{1}{\mathscr{C}_{n-1}}$$

for every $t \in \mathcal{B}_n$ and $n \geq 0$.

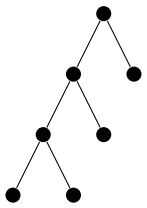
Another well-known leaf-centric binary tree source is the binomial random tree model [66], where the mapping ℓ corresponds to a binomial distribution:

Example 3.4. Fix a constant $p \in (0, 1)$ and define

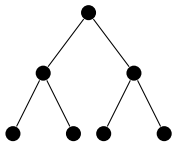
$$\ell_{bin,p}(i, n-i) = p^{i-1}(1-p)^{n-i-1}\binom{n-2}{i-1} \tag{3.3}$$

for every $n \geq 2$ and $1 \leq i \leq n-1$. This leaf-centric binary tree source corresponds to the *binomial random tree model*, which was studied in [66] for the case $p = 1/2$, and which is a slight variant of the *digital search tree model*, [27, 81].

Figure 3.2 shows an example of two full binary trees together with the probabilities assigned to them by the leaf-centric binary tree sources from Example 3.2, Example 3.3 and Example 3.4.



$P_{\ell_{bst}}(t_1) = \frac{1}{6}$
$P_{\ell_{uni}}(t_1) = \frac{1}{5}$
$P_{\ell_{bin,1/4}}(t_1) = \frac{1}{64}$



$P_{\ell_{bst}}(t_2) = \frac{1}{3}$
$P_{\ell_{uni}}(t_2) = \frac{1}{5}$
$P_{\ell_{bin,1/4}}(t_2) = \frac{3}{8}$

Figure 3.2: Two full binary trees t_1 (left) and t_2 (right), with the probabilities assigned to them by the binary search tree model, the uniform model and the binomial random tree model with $p = 1/4$.

In this chapter, we are interested in the expected number of *distinct fringe subtrees* in random full binary trees which are generated by leaf-centric binary tree sources. With $T_{n,\ell}$ we denote a random full binary tree of leafsize n drawn from the set $\mathcal{B}_n$ according to the probability mass function $P_\ell \colon \mathcal{B}_n \to [0,1]$ from (3.2), and with $F_{n,\ell}$ we denote the (random) number of distinct fringe subtrees occurring in the random tree $T_{n,\ell}$. In the following, we investigate $F_{n,\ell}$ under certain conditions on the mapping $\ell \in \mathcal{L}$. That is, we will assume that the mapping ℓ satisfies certain properties and then derive upper and lower bounds on $\mathbb{E}(F_{n,\ell})$ with respect to these properties. In particular, we consider four classes of leaf-centric binary tree sources, listed below, for which we will be able to derive asymptotic upper or lower bounds on $\mathbb{E}(F_{n,\ell})$. Here and in the rest of the chapter, increasing/decreasing functions are not necessarily strictly monotone, that is, if f is increasing, then $f(x) \geq f(y)$ whenever $x > y$, but not necessarily $f(x) > f(y)$ (and analogously for decreasing functions).

(i) A leaf-centric binary tree source with mapping ℓ is called *ψ-upper-bounded* for a decreasing function $\psi \colon \mathbb{R} \to (0,1]$ if there is a constant N such that $\ell(i, n-i) \leq \psi(n)$ for all $1 \leq i \leq n-1$ and $n \geq N$ (see Definition 3.8 in Section 3.3).

(ii) A leaf-centric binary tree source with mapping ℓ is called *φ-weakly-balanced* for a decreasing function $\varphi \colon \mathbb{N} \to (0,1]$ if there is a constant $\gamma \in (0, \frac{1}{2})$ and an integer N such that

$$\sum_{\gamma n \leq i \leq (1-\gamma)n} \ell(i, n-i) \geq \varphi(n)$$

for all $n \geq N$ (see Definition 3.14 in Section 3.4).

(iii) A leaf-centric binary tree source with mapping ℓ is *ς-strongly-balanced* for a decreasing function $\varsigma \colon \mathbb{R} \to (0,1]$ if there is a constant $\gamma \in (0, \frac{1}{2})$, an integer N and a constant C such that

$$\sum_{r \leq i \leq n-r} \ell(i, n-i) \geq \varsigma(r)$$

for all $n \geq N$ and $C \leq r \leq \lceil \gamma n \rceil$ (see Definition 3.19 in Section 3.5).

(iv) A leaf-centric binary tree source with mapping ℓ is called *ϱ-unbalanced* for a decreasing function ϱ if there is a constant $\gamma \in (0, \frac{1}{2})$, an integer N and a constant C such that

$$\sum_{r \le i \le n-r} \ell(i, n-i) \le \varrho(r)$$

for all $n \ge N$ and $C \le r \le \lceil \gamma n \rceil$ (see Definition 3.32 in Section 3.7).

Property (i) quantifies how "close" ℓ is to the binary search tree model. The latter is ψ-upper-bounded for $\psi(n) = 1/(n-1)$, which is the smallest function ψ for which property (i) is reasonable (since the sum over all $\ell(i, n-i)$ for $1 \le i \le n-1$ has to be one). Property (ii) generalizes the concept of balanced binary tree sources from [S3], [105]: When randomly constructing a binary tree with respect to a leaf-centric tree source of type (ii), the probability that the current weight is roughly equally split among the two children is bounded below by the function φ. Therefore, for slowly decreasing functions φ, balanced trees are preferred by this model. Property (iii) is a stronger constraint than property (ii): Every leaf-centric binary tree source that is ς-strongly-balanced for a function ς is also φ-weakly-balanced for the function φ defined by $\varphi(n) = \varsigma(\lceil \gamma n \rceil)$.

As our main results of this chapter, we obtain the following asymptotic bounds on the expected number of distinct fringe subtrees for these classes of leaf-centric binary tree sources. Let ℓ be a leaf-centric binary tree source.

(a) If ℓ is ψ-upper-bounded, then $\mathbb{E}(F_{n,\ell}) \le \mathcal{O}(n\psi(\log n))$ (Theorem 3.10, where a slightly more general result is shown).

(b) If ℓ is ψ-upper-bounded and there are constants $N \in \mathbb{N}$, $\kappa < 1$ such that $\psi(x) < \kappa$ for all $x \ge N$, then $\mathbb{E}(F_{n,\ell}) \ge \Omega\big(\frac{n}{\log n}\big)$ (Theorem 3.24).

(c) If ℓ is φ-weakly-balanced, then $\mathbb{E}(F_{n,\ell}) \le \mathcal{O}\big(\frac{n}{\varphi(n) \log n}\big)$ (Theorem 3.15).

(d) If ℓ is ς-strongly-balanced, then $\mathbb{E}(F_{n,\ell}) \le \mathcal{O}\big(\frac{n}{\varsigma(\log n) \log n}\big)$ (Theorem 3.20).

(e) If ℓ is ϱ-unbalanced and some additional technical conditions are satisfied, then $\mathbb{E}(F_{n,\ell}) \ge \Omega\big(\frac{n}{\varrho(\log n) \log n}\big)$ (Theorem 3.33).

The precise statements and asymptotic bounds including leading constants for the main term are given in the respective theorems. The upper bounds (a), (c) and (d) will be presented in Section 3.3, Section 3.4 and Section 3.5. The lower bounds (b) and (e) are then derived in Section 3.6 and Section 3.7.

These results are published in [S8, S9]. A first version [S8] of these results appeared in the Proceedings of MFCS 2018: The results (a), (b) and (c) already appeared in this conference version. A journal version [S9] of [S8] is currently under review. In the journal version, among others, several proofs needed for the respective results are simplified, leading constants in front of the main terms in the asymptotic bounds are improved and the results (d) and (e) are added. Furthermore, a preliminary weaker result of result (c) appeared in [S3].

These results generalize the results from [24, 36] on the binary search tree model in the following sense: from each of the results (a), (c) and (d) (respectively, (b) and (e)), we obtain the upper bound $\mathcal{O}(n/\log n)$ (respectively, the lower bound $\Omega(n/\log n)$) on the expected number of distinct fringe subtrees occurring in a random binary search tree of leafsize n. Furthermore, results (d) and (e) combined yield that the expected number of distinct fringe subtrees in a uniformly random full binary tree of leafsize n is in $\Theta(n/\sqrt{\log n})$, as shown in [38, 96].

Finally, results (b) and (c) imply the asymptotic bound $\Theta(n/\log n)$ on the expected number of distinct fringe subtrees in a random full binary tree of leafsize n generated according to the binomial random tree model presented in Example 3.4.

3.2 The cut-point argument

In order to obtain upper bounds on the average number of distinct fringe subtrees $\mathbb{E}(F_{n,\ell})$, we make use of a technique called *cut-point argument*, which was used before in several works for similar investigations (see for example [24, 36, 96]), and will also be applied for similar purposes in Chapter 4 and Chapter 5 of this work. The main idea of this cut-point technique is the following. Fix an integer $k > 0$. Let $t \in \mathcal{B}$ be a full binary tree. The number of distinct fringe subtrees occurring in t equals

(i) the number of distinct fringe subtrees of leafsize larger than k occurring in t plus

(ii) the number of distinct fringe subtrees of leafsize at most k occurring in t.

The number (i) of distinct fringe subtrees of leafsize larger than k is then bounded above by the number of *all fringe subtrees* in t of leafsize larger than k (for instance, the tree in Figure 3.1 (left) has 7 fringe subtrees of leafsize larger than one, but only 5 distinct fringe subtrees of leafsize larger than one). The number (ii) of distinct fringe subtrees of leafsize at most k is bounded above by the number of *all full binary trees* of leafsize at most k (irrespective of their occurrence in t). Note that this upper bound on number (ii) is a deterministic quantity; it is a sum of Catalan numbers. Let $Y_{n,k,\ell}$ denote the (random) number of (all) fringe subtrees of leafsize larger than k occurring in the random tree $T_{n,\ell}$ (if ℓ is clear from the context, we shortly write $Y_{n,k}$ for $Y_{n,k,\ell}$). In other words, $Y_{n,k,\ell}$ is the following random variable:

$$Y_{n,k,\ell} = |\{v \text{ node of } T_{n,\ell} \mid \|T_{n,\ell}[v]\| > k\}|. \tag{3.4}$$

Note that $Y_{n,k,\ell} = 0$ for all $\ell \in \mathcal{L}$ if $n \leq k$. Also, note that the definition of $Y_{n,k,\ell}$ also makes sense if k is not an integer. Using the cut-point argument, we obtain the following upper bound on $\mathbb{E}(F_{n,\ell})$.

Lemma 3.5. *Let $\ell \in \mathcal{L}$. The number $F_{n,\ell}$ of distinct fringe subtrees occurring in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \leq \mathbb{E}(Y_{n,k,\ell}) + \sum_{i=0}^{k-1} \mathscr{C}_i$$

with $Y_{n,k,\ell}$ as defined in (3.4) *and where $\mathscr{C}_i$ is the i^{th} Catalan number.*

Thus, in the following, we will focus on obtaining upper bounds on $\mathbb{E}(Y_{n,k,\ell})$ with respect to certain conditions on ℓ in order to obtain upper bounds on $\mathbb{E}(F_{n,\ell})$. The integer k is called the *cut-point*; it has to be chosen in a suitable way in order to obtain suitable upper bounds on $\mathbb{E}(F_{n,\ell})$. The following notation will be useful: for a function $\ell \in \mathcal{L}$, we define $\ell^* \colon \mathbb{N} \times \mathbb{N} \to [0,1]$ as

$$\ell^*(i,j) = \begin{cases} \ell(i,j) + \ell(j,i) & \text{if } i \neq j, \\ \ell(i,j) & \text{if } i = j. \end{cases} \tag{3.5}$$

Note that $\ell^*(i,j) \leq 1$ for all $i,j \in \mathbb{N}$ and that $\sum_{1 \leq i \leq n/2} \ell^*(i, n-i) = 1$ (where the sum goes over all integers in the interval $[1, n/2]$).

First, we observe that $\mathbb{E}(Y_{n,k,\ell})$ satisfies the following recurrence relation.

Lemma 3.6. *Let $\ell \in \mathcal{L}$, and let $n > k \geq 0$. Then the number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{1 \leq i \leq n/2} \ell^*(i, n-i) \left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right). \tag{3.6}$$

Moreover, if $k+1 > n/2$ we have

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{i=k+1}^{n-1} \ell^*(i, n-i)\mathbb{E}(Y_{i,k}), \tag{3.7}$$

and if $k+1 \leq n/2$ we have

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{i=k+1}^{n-k-1} \ell(i, n-i) \left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right) + \sum_{i=n-k}^{n-1} \ell^*(i, n-i)\mathbb{E}(Y_{i,k}). \tag{3.8}$$

Proof. Let $t \in \mathcal{B}_n$ be a full binary tree. Then the number of fringe subtrees of leafsize larger than k occurring in t equals the number of fringe subtrees of leafsize larger than k occurring in its left subtree t_l plus the number of fringe subtrees of leafsize larger than k occurring in the right subtree t_r plus one (for the tree itself, i.e., the fringe subtree rooted in the root node of t). As the left and right subtree of a random tree $T_{n,\ell}$ conditioned on their leafsizes i and $n-i$ for some integer $1 \leq i \leq n-1$ are again independent random trees $T_{i,\ell}$ and $T_{n-i,\ell}$, and as the probability that the left and right subtree are of leafsizes i

and $n-i$ is given by $\ell(i, n-i)$, we find

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{i=1}^{n-1} \ell(i, n-i) \left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right).$$

From this and the definition of ℓ^* (see (3.5)) we obtain (3.6). With $\mathbb{E}(Y_{i,k}) = 0$ for $i \leq k$ we can write (3.6) as

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{k+1 \leq i \leq n/2} \ell^*(i, n-i)\mathbb{E}(Y_{i,k}) + \sum_{1 \leq i \leq \min\{n-(k+1), n/2\}} \ell^*(i, n-i)\mathbb{E}(Y_{n-i,k}).$$

By considering the case $k+1 > n/2$ and $k+1 \leq n/2$, we finally obtain (3.7) and (3.8). □

In this chapter, several times we have to bound expressions of the form $\sum_{1 \leq i \leq n/2} \ell^*(i, n-i) \cdot \min\{\alpha i, \beta\}$ for constants $\alpha, \beta > 0$. It turns out to be convenient to use the (cumulative) distribution function $D_{\ell,n} \colon \mathbb{R}_{\geq 0} \to \mathbb{R}$ corresponding to ℓ^*, which is defined as follows.

$$D_{\ell,n}(x) = \sum_{i \leq x} \ell^*(i, n-i). \tag{3.9}$$

The sum goes over all integers i with $1 \leq i \leq \min\{n/2, x\}$. If ℓ is clear from the context, we also write D_n for $D_{\ell,n}$. In the following lemma we use a Riemann–Stieltjes integral; see e.g. [99, Chapter 6].

Lemma 3.7. *Let $\ell \in \mathcal{L}$ and $\alpha, \beta > 0$. Then we have*

$$\sum_{1 \leq i \leq n/2} \ell^*(i, n-i) \cdot \min\{\alpha i, \beta\} = \beta - \int_0^{\beta/\alpha} \alpha D_n(x) \mathrm{d}x.$$

Proof. Using integration by parts for Riemann–Stieltjes integrals (see for example [99, p. 141]), we obtain

$$\begin{aligned}
\sum_{1 \leq i \leq \frac{n}{2}} \ell^*(i, n-i) \cdot \min\{\alpha i, \beta\} &= \int_0^{\infty} \min\{\alpha x, \beta\} \, \mathrm{d}D_n(x) \\
&= \int_0^{\beta/\alpha} \alpha x \, \mathrm{d}D_n(x) + \int_{\beta/\alpha}^{\infty} \beta \, \mathrm{d}D_n(x) \\
&= \Big[\alpha x D_n(x)\Big]_0^{\beta/\alpha} - \int_0^{\beta/\alpha} \alpha D_n(x) \, \mathrm{d}x + \Big[\beta D_n(x)\Big]_{\beta/\alpha}^{\infty} \\
&= \beta - \int_0^{\beta/\alpha} \alpha D_n(x) \mathrm{d}x.
\end{aligned}$$

This proves the lemma. □

3.3 Upper-bounded sources

The first natural class of leaf-centric tree sources we consider is the following class, where the mapping ℓ (resp. ℓ^*) is bounded from above by a function ψ.

Definition 3.8 (ψ-upper-bounded sources)**.** Let $\psi\colon \mathbb{R} \to (0,1]$ be a decreasing function. With $\mathcal{L}_{up}(\psi) \subseteq \mathcal{L}$ we denote the set of mappings $\ell \in \mathcal{L}$ for which there is an integer N_ℓ such that

$$\ell(i, n-i) \leq \psi(n)$$

for all $n \geq N_\ell$ and all integers $1 \leq i \leq n-1$. In the same way, let $\mathcal{L}^*_{up}(\psi) \subseteq \mathcal{L}$ denote the set of mappings $\ell \in \mathcal{L}$ for which there is an integer N_ℓ such that

$$\ell^*(i, n-i) \leq \psi(n)$$

for all $n \geq N_\ell$ and all integers $1 \leq i \leq n-1$.

Note that by definition of ℓ^* (see (3.5)), we have $\mathcal{L}^*_{up}(\psi) \subseteq \mathcal{L}_{up}(\psi) \subseteq \mathcal{L}^*_{up}(2\psi)$. Also note that without loss of generality, we can assume that $\psi(n) \geq 1/(n-1)$ for every integer $n \geq 2$, as the values $\ell(i, n-i)$ have to add up to 1 for $1 \leq i \leq n-1$ by condition (3.1). In the same way, if we consider the class $\mathcal{L}^*_{up}(\psi)$, we can assume that $\psi(n) \geq 2/n$ for every integer $n \geq 2$. The class of ψ-upper-bounded functions can be generalized as follows:

Definition 3.9 (ψ-weakly-upper-bounded sources)**.** For a decreasing function $\psi\colon \mathbb{R} \to (0,1]$, let $\mathcal{L}_{wup}(\psi) \subseteq \mathcal{L}$ denote the set of mappings $\ell \in \mathcal{L}$ for which there is an integer N_ℓ such that

$$\sum_{1 \leq i \leq k} \ell^*(i, n-i) \leq k\psi(n)$$

for all $n \geq N_\ell$ and all integers $1 \leq k \leq n/2$.

Note that we have $\mathcal{L}^*_{up}(\psi) \subseteq \mathcal{L}_{wup}(\psi)$ and hence also $\mathcal{L}_{up}(\psi) \subseteq \mathcal{L}_{wup}(2\psi)$. For ψ-weakly-upper-bounded sources we can assume again that $\psi(n) \geq 2/n$ for every integer $n \geq 2$, as the values $\ell^*(i, n-i)$ have to add up to 1 for $1 \leq i \leq n/2$ by condition (3.1). As our first main theorem of this chapter, we obtain the following upper bound on $\mathbb{E}(F_{n,\ell})$ for mappings $\ell \in \mathcal{L}_{wup}(\psi)$:

Theorem 3.10. *Let $\psi\colon \mathbb{R} \to (0,1]$ be a decreasing function, and let $\ell \in \mathcal{L}_{wup}(\psi)$. The number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \leq 2n\psi(\log_4 n) + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}}\right).$$

As $\mathcal{L}_{up}(\psi) \subseteq \mathcal{L}^*_{up}(2\psi)$ and $\mathcal{L}^*_{up}(\psi) \subseteq \mathcal{L}_{wup}(\psi)$, we immediately obtain an asymptotic upper bound on $\mathbb{E}(F_{n,\ell})$ for $\ell \in \mathcal{L}_{up}(\psi)$ and $\ell \in \mathcal{L}^*_{up}(\psi)$ from Theorem 3.10 as well. In order to prove Theorem 3.10, we make use of the cut-point technique as described in Section 3.2. For this, we start with an upper

bound on the expectation $\mathbb{E}(Y_{n,k,\ell})$ of fringe subtrees of leafsize larger than k in a random tree $T_{n,\ell}$, where $\ell \in \mathcal{L}_{wup}(\psi)$ for some mapping ψ.

Lemma 3.11. *Let $\psi\colon \mathbb{R} \to (0,1]$ be a decreasing function, and let $\ell \in \mathcal{L}_{wup}(\psi)$ with integer N_ℓ. Then the (random) number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,k}) \leq 2n\psi(k) - 2$$

for all n, k with $n > k \geq N_\ell$.

Proof. As $\ell \in \mathcal{L}_{wup}(\psi)$, we assume that $1 \geq \psi(n) \geq 2/n$ for every integer $n \geq N_\ell$. For the distribution function $D_n = D_{\ell,n}$ in (3.9), we have

$$D_n(x) \leq x\psi(n) \tag{3.10}$$

for all $x \geq 0$. We prove the statement by induction on $n > k \geq N_\ell$. For the base case, let $n = k+1$. As there is exactly one fringe subtree of leafsize larger than k in a full binary tree of leafsize $k+1$ (the fringe subtree rooted at the root node), we have

$$\mathbb{E}(Y_{k+1,k}) = 1 \leq 2(k+1)\psi(k) - 2,$$

as $\psi(k) \geq 2/k$ by assumption. For the induction step, we take an integer $n > k+1 > N_\ell$, so that $\mathbb{E}(Y_{i,k}) \leq 2i\psi(k) - 2$ for every $k < i \leq n-1$. Lemma 3.6 yields

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{1 \leq i \leq n/2} \ell^*(i, n-i)\left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right).$$

Note that $\mathbb{E}(Y_{i,k}) \leq \max\{2i\psi(k) - 2, 0\}$ for all $1 \leq i \leq n/2$: if $i > k$, we find that $\mathbb{E}(Y_{i,k}) \leq 2i\psi(k) - 2$ holds by the induction hypothesis, and otherwise, if $i \leq k$, we trivially have $\mathbb{E}(Y_{i,k}) = 0$. Furthermore, we have $\mathbb{E}(Y_{n-i,k}) \leq 2(n-i)\psi(k) - 2$ for all $1 \leq i \leq n/2$: if $n - i > k$, this holds by the induction hypothesis, and if $n - i \leq k$, we find $\mathbb{E}(Y_{n-i,k}) = 0 < \frac{2n}{k} - 2 \leq n\psi(k) - 2 \leq 2(n-i)\psi(k) - 2$. We can combine these to

$$\begin{aligned}\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k}) &\leq \max\{2i\psi(k) - 2, 0\} + 2(n-i)\psi(k) - 2 \\ &= 2n\psi(k) - 2 - \min\{2i\psi(k), 2\}.\end{aligned}$$

Thus, we obtain

$$\begin{aligned}\mathbb{E}(Y_{n,k}) &\leq 1 + \sum_{1 \leq i \leq n/2} \ell^*(i, n-i)(2n\psi(k) - 2 - \min\{2i\psi(k), 2\}) \\ &= 2n\psi(k) - 1 - \sum_{1 \leq i \leq n/2} \ell^*(i, n-i)\min\{2i\psi(k), 2\} \\ &= 2n\psi(k) - 3 + \int_0^{1/\psi(k)} 2\psi(k) D_n(x)\,\mathrm{d}x,\end{aligned}$$

where the last equation follows from Lemma 3.7. Using (3.10), we obtain

$$\begin{aligned}\mathbb{E}(Y_{n,k}) &\leq 2n\psi(k) - 3 + \int_0^{1/\psi(k)} 2\psi(k)\psi(n)x \,\mathrm{d}x \\ &= 2n\psi(k) - 3 + \Big[\psi(k)\psi(n)x^2\Big]_0^{1/\psi(k)} \\ &= 2n\psi(k) - 3 + \frac{\psi(n)}{\psi(k)} \\ &\leq 2n\psi(k) - 2,\end{aligned}$$

where we use the fact that ψ is decreasing and $n > k$ for the last inequality. □

We are now able to prove Theorem 3.10:

Proof of Theorem 3.10. Let $\ell \in \mathcal{L}_{wup}(\psi)$ with integer N_ℓ. Let $n > 4^{N_\ell}$, and set $k := \lceil \log_4 n \rceil > N_\ell$. We use the cut-point argument from Lemma 3.5 together with Lemma 3.11 to obtain

$$\mathbb{E}(F_{n,\ell}) \leq \mathbb{E}(Y_{n,k}) + \sum_{i=0}^{k-1} \mathscr{C}_i \leq 2n\psi(k) + \sum_{i=0}^{k-1} \mathscr{C}_i.$$

With the asymptotic growth of the Catalan numbers (2.1), we obtain

$$\mathbb{E}(F_{n,\ell}) \leq 2n\psi(k) + \mathcal{O}\left(\frac{4^k}{k^{3/2}}\right) \leq 2n\psi(\log_4 n) + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}}\right),$$

as $k = \lceil \log_4 n \rceil$. This yields the asymptotic upper bound from Theorem 3.10. □

Using Theorem 3.10, we obtain an upper bound on the expected number of distinct fringe subtrees in random binary search trees:

Example 3.12. For the binary search tree model from Example 3.2, we find that $\ell_{bst} \in \mathcal{L}^*_{up}(\psi_{bst})$, where $\psi_{bst} \colon \mathbb{R} \to (0,1]$ is given by $\psi_{bst}(x) = 2/(x-1)$. Theorem 3.10 gives us the following upper bound on the expected number of distinct fringe subtrees in a random binary search tree:

$$\mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{8n}{\log n}\left(1 + o(1)\right).$$

Recall that [36, 24] proved that

$$\mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{4n}{\log n}\left(1 + o(1)\right),$$

and that [24] also showed a lower bound

$$\mathbb{E}(F_{n,\ell_{bst}}) \geq \frac{\log(3)n}{2\log n}\left(1 + o(1)\right).$$

Thus, except for the leading constant, the upper bound from Theorem 3.10 yields the correct asymptotic growth of $\mathbb{E}(F_{n,\ell_{bst}})$. In Chapter 5 in Corollary 5.6, we show that

$$\frac{c_1 n}{\log n}(1+o(1)) \leq \mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{4n}{\log n}(1+o(1)), \tag{3.11}$$

where $c_1 \approx 3.472754274$.

For the binomial random tree model, we are also able to obtain a non-trivial upper bound on $\mathbb{E}(F_{n,\ell_{bin,p}})$ from Theorem 3.10:

Example 3.13. For the binomial random tree model from Example 3.4, a short computation shows that for a fixed integer n, the maximal value of $\ell_{bin,p}(i, n-i)$ with $1 \leq i \leq n-1$ is attained at $i_{\max}(n) = \lfloor p(n-1) \rfloor + 1$. In particular, we have

$$\ell_{bin,p}(i_{\max}(n), n - i_{\max}(n)) = \Theta(1/\sqrt{n}).$$

Thus, we find that $\ell_{bin,p} \in \mathcal{L}^*_{up}(\psi_{bin,p})$ for a function $\psi_{bin,p}$ which satisfies $\psi_{bin,p}(x) = \Theta(1/\sqrt{x})$. By Theorem 3.10, we obtain $\mathbb{E}(F_{n,\ell_{bin,p}}) \leq \mathcal{O}(n/\sqrt{\log n})$. However, in the following sections, we will be able to prove the stronger upper bound $\mathbb{E}(F_{n,\ell_{bin,p}}) \leq \mathcal{O}(n/\log n)$ (see Example 3.18) and also a corresponding lower bound $\mathbb{E}(F_{n,\ell_{bin,p}}) \geq \Omega(n/\log n)$ (see Example 3.30).

There are plenty of other ways to choose the mapping ψ in order to obtain non-trivial upper bounds on $\mathbb{E}(F_{n,\ell})$ for $\ell \in \mathcal{L}^*_{up}(\psi)$: For example, $\psi(x) = \Theta(x^{-\alpha})$ for a constant $0 < \alpha \leq 1$ yields $\mathbb{E}(F_{n,\ell}) \leq \mathcal{O}(n/(\log n)^{\alpha})$, and $\psi(x) = \Theta(1/\log x)$ yields $\mathbb{E}(F_{n,\ell}) \leq \mathcal{O}(n/\log\log n)$. Note that Theorem 3.10 only makes a non-trivial statement if $\psi(n) < 1/2$ for $n \geq N_\ell$. For the uniform probability distribution (see Example 3.3), Theorem 3.10 only yields an upper bound of the form $\mathbb{E}(F_{n,\ell_{uni}}) \leq \mathcal{O}(n)$, as we have $\ell_{uni}(1, n-1) > 1/4$ for every $n \geq 2$.

3.4 Weakly-balanced sources

In this section, we investigate another class of leaf-centric binary tree sources for which we will show an upper bound on $\mathbb{E}(F_{n,\ell})$. We consider *weakly-balanced* binary tree sources, which represent a generalization of balanced binary tree sources introduced in [105] and further analyzed in [S3].

Definition 3.14 (φ-weakly-balanced sources)**.** Let $\varphi\colon \mathbb{N} \to (0,1]$ be a decreasing function and let $\gamma \in (0, \frac{1}{2})$. With $\mathcal{L}_{wbal}(\varphi, \gamma) \subseteq \mathcal{L}$ we denote the set of mappings $\ell \in \mathcal{L}$ for which there is an integer N_ℓ such that

$$\sum_{\gamma n \leq i \leq (1-\gamma)n} \ell(i, n-i) \geq \varphi(n)$$

for all $n \geq N_\ell$.

For the class of weakly-balanced leaf-centric tree sources, we obtain the following main result:

Theorem 3.15. *Let $\varphi\colon \mathbb{N} \to (0,1]$ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$, and let $\ell \in \mathcal{L}_{wbal}(\varphi, \gamma)$. The number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \leq \frac{n}{\gamma\varphi(n)\log_4 n} + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}}\right).$$

In order to get a nontrivial statement from Theorem 3.15, we should have $\varphi(n) \geq \Omega(1/\log n)$. We start with the following lemma:

Lemma 3.16. *Let $\varphi\colon \mathbb{N} \to (0,1]$ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$, and let $\ell \in \mathcal{L}_{wbal}(\varphi, \gamma)$ with integer N_ℓ. Then for $k \geq N_\ell$ and $n \geq k+1$, the (random) number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,k}) \leq \frac{n}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)}.$$

Proof. We prove the statement by induction on $n \geq k+1$. For the base case, let $n = k+1$. As there is exactly one fringe subtree of leafsize larger than k in a full binary tree of leafsize $k+1$ (the fringe subtree rooted at the root node), we have

$$\mathbb{E}(Y_{k+1,k}) = 1 \leq \frac{(\frac{1}{\gamma} - 1)(k+1)}{\varphi(k+1)k} \leq \frac{k+1}{\gamma\varphi(k+1)k} - \frac{1}{\varphi(k+1)},$$

as $\frac{1}{\gamma} > 2$ and $\varphi(k+1) \leq 1$ by assumption. For the induction step, take an integer $n > k+1$, so that

$$\mathbb{E}(Y_{i,k}) \leq \frac{i}{\gamma\varphi(i)k} - \frac{1}{\varphi(i)} \tag{3.12}$$

for every integer $k+1 \leq i \leq n-1$. As $\gamma k < \gamma n$ and $\ell \in \mathcal{L}_{wbal}(\varphi, \gamma)$, we have

$$D_n(x) \leq 1 - \varphi(n) \tag{3.13}$$

for every $0 \leq x \leq \gamma k$. Lemma 3.6 yields

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{1 \leq i \leq n/2} \ell^*(i, n-i)\left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right).$$

First, we observe that for all $1 \leq i \leq n/2$, we have

$$\mathbb{E}(Y_{i,k}) \leq \max\left\{\frac{i}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)}, 0\right\}.$$

To see this, note that (i) $\mathbb{E}(Y_{i,k}) = 0$ for $i \leq k$, and (ii) (3.12) holds for $i > k$ and φ is decreasing. Furthermore, for all $1 \leq i \leq n/2$ we have

$$\mathbb{E}(Y_{n-i,k}) \leq \frac{n-i}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)}.$$

If $n - i > k$, this follows from (3.12) and the fact that φ is decreasing. On the other hand, if $n - i \leq k$, then we have

$$\mathbb{E}(Y_{n-i,k}) = 0 < \frac{n - 2\gamma k}{2\gamma\varphi(n)k} = \frac{n}{2\gamma\varphi(n)k} - \frac{1}{\varphi(n)} \leq \frac{n-i}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)},$$

as $\gamma < 1/2$ by assumption. Thus, we obtain

$$\begin{aligned}
\mathbb{E}(Y_{n,k}) &\leq 1 + \sum_{1 \leq i \leq \frac{n}{2}} \ell^*(i, n-i)\Big(\max\left\{ \frac{i}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)}, 0 \right\} + \frac{n-i}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)} \Big) \\
&= 1 + \sum_{1 \leq i \leq \frac{n}{2}} \ell^*(i, n-i)\Big(\frac{n}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)} - \min\left\{ \frac{i}{\gamma\varphi(n)k}, \frac{1}{\varphi(n)} \right\} \Big) \\
&= 1 + \frac{n}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)} - \sum_{1 \leq i \leq \frac{n}{2}} \ell^*(i, n-i) \min\left\{ \frac{i}{\gamma\varphi(n)k}, \frac{1}{\varphi(n)} \right\} \\
&\overset{\text{Lem. 3.7}}{=} 1 + \frac{n}{\gamma\varphi(n)k} - \frac{2}{\varphi(n)} + \int_0^{\gamma k} \frac{D_n(x)}{\gamma\varphi(n)k} \, \mathrm{d}x \\
&\overset{(3.13)}{\leq} 1 + \frac{n}{\gamma\varphi(n)k} - \frac{2}{\varphi(n)} + \int_0^{\gamma k} \frac{1 - \varphi(n)}{\gamma\varphi(n)k} \, \mathrm{d}x \\
&= \frac{n}{\gamma\varphi(n)k} - \frac{1}{\varphi(n)},
\end{aligned}$$

which concludes the proof. □

With Lemma 3.16, we are now able to prove Theorem 3.15.

Proof of Theorem 3.15. Let $\ell \in \mathcal{L}_{wbal}(\varphi, \gamma)$ with integer N_ℓ. Let $n > 4^{N_\ell}$, and set $k := \lceil \log_4 n \rceil > N_\ell$. Again, we use Lemma 3.5 to obtain

$$\mathbb{E}(F_{n,\ell}) \leq \mathbb{E}(Y_{n,k}) + \sum_{i=0}^{k-1} \mathscr{C}_i.$$

With Lemma 3.16, and the asymptotic growth of the Catalan numbers, we find

$$\mathbb{E}(F_{n,\ell}) \leq \frac{n}{\gamma\varphi(n)k} + \sum_{i=0}^{k-1} \mathscr{C}_i \leq \frac{n}{\gamma\varphi(n)\log_4 n} + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}} \right).$$

This finishes the proof. □

The application of Theorem 3.15 yields the following upper bound on the expected number of distinct fringe subtrees in random binary search trees:

Example 3.17. For the binary search tree model from Example 3.2, we find that $\ell_{bst} \in \mathcal{L}_{wbal}(\varphi_{bst}, \gamma_{bst})$ with $\varphi_{bst}(n) = 1/2$ for every $n \geq 2$ and $\gamma_{bst} = 1/4$. Hence, by Theorem 3.15, we have

$$\mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{16n}{\log n} + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}} \right).$$

The asymptotic growth coincides with the upper bound on $\mathbb{E}(F_{n,\ell_{bst}})$ from Example 3.12 obtained by Theorem 3.10, except for the leading constant. Recall the upper and lower bound (3.11) for the binary search tree model (where the upper bound was already shown in [36, 24]). Thus, except for the leading constant, the upper bound from Theorem 3.15 yields the correct asymptotic growth of $\mathbb{E}(F_{n,\ell_{bst}})$ as well. Note that it is possible to choose $\varphi_{bst}(n) = 1 - 2\gamma$ for any $\gamma < 1/2$. For $\gamma = 1/4$, we obtain the best leading constant in the upper bound.

Example 3.18. For the binomial random tree model from Example 3.4, we obtain the following result from Theorem 3.15. Let S_p^n be the random variable taking values in the set $\{1, \ldots, n-1\}$ according to the probability mass function $\ell_{bin,p}$ from Example 3.4. Then $S_p^n = \mathrm{Bin}(n-2, p) + 1$, where $\mathrm{Bin}(n-2, p)$ is a binomially distributed random variable with parameters $n-2$ and p. For the expected value μ of S_p^n, we thus obtain $\mu = \mathbb{E}(S_p^n) = p(n-2) + 1$. Chernoff's bound implies

$$\mathbb{P}(|\mu - S_p^n| < \mu^{3/4}) \geq 1 - 2e^{-\frac{\sqrt{\mu}}{3}}.$$

Moreover, with $i_1 := \mu - \mu^{3/4}$ and $i_2 := \mu + \mu^{3/4}$ we have

$$\mathbb{P}(|\mu - S_p^n| < \mu^{3/4}) = \sum_{i_1 < i < i_2} \ell_{bin,p}(i, n-i).$$

Next, let $\gamma < \min\{p, 1-p\} \leq 1/2$. There is an integer N_ℓ (depending on p and γ) such that

$$\gamma n \leq \mu - \mu^{3/4} \leq \mu + \mu^{3/4} \leq (1-\gamma)n$$

for all $n \geq N_\ell$. All in all, we thus have

$$\sum_{\gamma n \leq i \leq n - \gamma n} \ell_{bin,p}(i, n-i) \geq \sum_{i_1 < i < i_2} \ell_{bin,p}(i, n-i) \geq 1 - 2e^{-\frac{\sqrt{p(n-2)+1}}{3}}$$

for $n \geq N_\ell$. Choose any constant ε with $0 < \varepsilon < 1$, and let $\varphi(n) = 1 - \varepsilon$ (i.e., φ is a constant function). Then, if n is large enough, we have

$$\sum_{\gamma n \leq i \leq n - \gamma n} \ell_{bin,p}(i, n-i) \geq 1 - \varepsilon,$$

i.e., $\ell_{bin,p} \in \mathcal{L}_{wbal}(\varphi, \gamma)$ for N_ℓ large enough. From Theorem 3.15, we obtain

$$\mathbb{E}(F_{n,\ell_{bin,p}}) \leq c_p \cdot \frac{n}{\log n} + \mathcal{O}\left(\frac{n}{(\log n)^{3/2}}\right),$$

where we can choose any constant c_p satisfying $c_p > 2/p$, if $p \leq 1/2$, respectively, $c_p > 2/(1-p)$, if $p > 1/2$, as $\varepsilon > 0$ is arbitrary. In particular, Theorem 3.15 yields a more precise upper bound on $\mathbb{E}(F_{n,\ell_{bin,p}})$ than Theorem 3.10 (see

Example 3.13). In Example 3.30, we will show a corresponding lower bound of the form $\mathbb{E}(F_{n,\ell_{bin,p}}) \geq \Omega(n/\log n)$.

It remains to remark that for the uniform probability distribution (from Example 3.3), Theorem 3.15 only yields a trivial upper bound on $\mathbb{E}(F_{n,\ell_{uni}})$. On the one hand, we would need $\varphi(n) \geq \omega(1/\log n)$ in order to obtain a non-trivial upper bound from Theorem 3.15, but on the other hand, for every constant $0 < \gamma < 1/2$, we find that if $\ell_{uni} \in \mathcal{L}_{wbal}(\varphi, \gamma)$, then $\varphi(n) \leq \mathcal{O}(1/\sqrt{n})$. In the next section, we present a class of leaf-centric binary tree sources which contains the uniform distribution ℓ_{uni} from Example 3.3, and for which we will be able to derive a non-trivial asymptotic upper bound on $\mathbb{E}(F_{n,\ell})$.

3.5 Strongly-balanced sources

In this section, we focus on another class of leaf-centric binary tree sources, which represents a refinement of the class of weakly-balanced leaf-centric tree sources from Definition 3.14 from the previous section.

Definition 3.19 (ς-strongly-balanced sources)**.** Let $\varsigma\colon \mathbb{R} \to (0,1]$ be a decreasing function and let $\gamma \in (0, \frac{1}{2})$. With $\mathcal{L}_{sbal}(\varsigma, \gamma) \subseteq \mathcal{L}$ we denote the set of mappings $\ell \in \mathcal{L}$ for which there is an integer N_ℓ and a constant $c_\ell \geq 1$ such that for every $n \geq N_\ell$ and every integer r with $c_\ell \leq r \leq \lceil \gamma n \rceil$, the following inequality holds:

$$\sum_{r \leq i \leq n-r} \ell(i, n-i) \geq \varsigma(r).$$

Let $\ell \in \mathcal{L}_{sbal}(\varsigma, \gamma)$, and define $\varphi\colon \mathbb{N} \to (0,1]$ by $\varphi(n) = \varsigma(\lceil \gamma n \rceil)$. Then we have $\ell \in \mathcal{L}_{wbal}(\varphi, \gamma)$, i.e., every strongly-balanced leaf-centric tree source is also weakly-balanced. In particular, Theorem 3.15 thus holds for strongly balanced tree sources as well.

However, we are able to prove a stronger asymptotic upper bound on the expected number $\mathbb{E}(F_{n,\ell})$ of distinct fringe subtrees in a random tree $T_{n,\ell}$ if ℓ corresponds to a strongly-balanced tree source.

Theorem 3.20. *Let $\varsigma\colon \mathbb{R} \to (0,1]$ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$, and let $\ell \in \mathcal{L}_{sbal}(\varsigma, \gamma)$. The number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \leq \frac{n}{\gamma \varsigma(\gamma \log_4 n) \log_4 n}(1 + o(1)).$$

In order to prove Theorem 3.20, we start again with an upper bound on the expected number $\mathbb{E}(Y_{n,k})$ of fringe subtrees of leafsize larger than k in a random tree $T_{n,\ell}$, where the leaf-centric tree source ℓ is contained in the set ouf sources $\mathcal{L}_{sbal}(\varsigma, \gamma)$ for some decreasing function ς and some number γ.

The following lemma and its proof are in fact quite similar to Lemma 3.16 and its proof.

Lemma 3.21. *Let $\varsigma\colon \mathbb{R} \to (0,1]$ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$, and let $\ell \in \mathcal{L}_{sbal}(\varsigma, \gamma)$ with integer N_ℓ and constant c_ℓ. Then for $k \geq \max\{N_\ell, c_\ell/\gamma\}$ and $n \geq k+1$, the (random) number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,k}) \leq \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}.$$

Proof. We prove the statement by induction on $n \geq k+1$. For the base case, let $n = k+1$. A full binary tree t of leafsize $k+1$ has exactly one fringe subtree of leafsize larger than k, thus

$$\mathbb{E}(Y_{k+1,k}) = 1 \leq \frac{(\frac{1}{\gamma}-1)(k+1)}{\varsigma(\lceil\gamma(k+1)\rceil)k} \leq \frac{k+1}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)},$$

as $\frac{1}{\gamma} > 2$ and $\varsigma(\lceil\gamma(k+1)\rceil) \leq 1$ by assumption. Let us now deal with the induction step. Take an integer $n > k+1$, so that

$$\mathbb{E}(Y_{i,k}) \leq \frac{i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)} \tag{3.14}$$

for every integer $k+1 \leq i \leq n-1$ by the induction hypothesis. Since we have $c_\ell \leq \gamma k < \lceil\gamma(k+1)\rceil \leq \lceil\gamma n\rceil$ and $\ell \in \mathcal{L}_{sbal}(\varsigma, \gamma)$, we find

$$D_n(x) \leq 1 - \varsigma(\lceil\gamma(k+1)\rceil) \tag{3.15}$$

for every $0 \leq x \leq \gamma k$ (take $r = \lceil\gamma(k+1)\rceil$ in Definition 3.19). By Lemma 3.6, we have

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{1\leq i\leq n/2} \ell^*(i, n-i)(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})).$$

First, note that for all $1 \leq i \leq n/2$ we have

$$\mathbb{E}(Y_{i,k}) \leq \max\left\{\frac{i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}, 0\right\}.$$

This follows from (3.14) for $i > k$, and from the fact that $\mathbb{E}(Y_{i,k}) = 0$ for $i \leq k$. Furthermore, for all $1 \leq i \leq n/2$ we have

$$\mathbb{E}(Y_{n-i,k}) \leq \frac{n-i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}.$$

This follows again from (3.14) if $n - i > k$. For $n - i \leq k$ note that

$$\begin{aligned}\mathbb{E}(Y_{n-i,k}) = 0 &< \frac{n-2\gamma k}{2\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} = \frac{n}{2\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)} \\ &\leq \frac{n-i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)},\end{aligned}$$

since $i \leq n/2$ and $\gamma < 1/2$ by assumption. Thus, we obtain

$$\begin{aligned}
\mathbb{E}(Y_{n,k}) &\leq 1 + \sum_{1\leq i\leq n/2} \ell^*(i,n-i)\Bigg(\max\left\{ \frac{i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}, 0\right\} \\
&\qquad + \frac{n-i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}\Bigg) \\
&= 1 + \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)} \\
&\qquad - \sum_{1\leq i\leq n/2} \ell^*(i,n-i) \min\left\{ \frac{i}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k}, \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}\right\} \\
&\overset{\text{Lem. 3.7}}{=} 1 + \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{2}{\varsigma(\lceil\gamma(k+1)\rceil)} + \int_0^{\gamma k} \frac{D_n(x)}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k}\,\mathrm{d}x \\
&\overset{(3.15)}{\leq} 1 + \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{2}{\varsigma(\lceil\gamma(k+1)\rceil)} + \int_0^{\gamma k} \frac{1-\varsigma(\lceil\gamma(k+1)\rceil)}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k}\,\mathrm{d}x \\
&= \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)},
\end{aligned}$$

which concludes the proof. □

Proof of Theorem 3.20. We are now able to prove Theorem 3.20. First, we set $m = \max\{N_\ell, c_\ell/\gamma\}$. By Lemma 3.21, the expected number $\mathbb{E}(Y_{n,k})$ of fringe subtrees of leafsize larger than k in $T_{n,\ell}$ satisfies

$$\mathbb{E}(Y_{n,k}) \leq \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} - \frac{1}{\varsigma(\lceil\gamma(k+1)\rceil)}$$

for all $n > k \geq m$. Now we set $k := \lfloor \log_4 n - 1/\gamma \rfloor - 1$ (which implies that $\varsigma(\lceil\gamma(k+1)\rceil) \geq \varsigma(\gamma \log_4 n)$), and let n be sufficiently large, so that $k \geq m$. By the cut-point argument from Lemma 3.5, we find that

$$\mathbb{E}(F_{n,\ell}) \leq \mathbb{E}(Y_{n,k}) + \sum_{i=0}^{k-1} \mathscr{C}_i.$$

With the asymptotic growth of the Catalan numbers (2.1), and with Lemma 3.21, we find

$$\begin{aligned}
\mathbb{E}(F_{n,\ell}) &\leq \frac{n}{\gamma\varsigma(\lceil\gamma(k+1)\rceil)k} + \mathcal{O}\left(\frac{4^k}{k^{3/2}}\right) \\
&\leq \frac{n}{\gamma\varsigma(\gamma\log_4 n)\log_4 n}(1+o(1)).
\end{aligned}$$

This finishes the proof. □

With Theorem 3.20, we obtain a non-trivial asymptotic upper bound on the expected number of distinct fringe subtrees in uniformly random binary tree.

Example 3.22. Consider the uniform distribution corresponding to ℓ_{uni} from Example 3.3. Let $0 < \gamma < 1/2$, and let $\varepsilon > 0$. From the asymptotic bound of the Catalan numbers (2.1), we find that there is an integer N_ε (depending on ε) such that

$$\frac{(1-\varepsilon)4^n}{\sqrt{\pi}(n+1)^{3/2}} \leq \mathscr{C}_n \leq \frac{(1+\varepsilon)4^n}{\sqrt{\pi}(n+1)^{3/2}}$$

for all $n \geq N_\varepsilon$. Let $N_\varepsilon \leq r \leq \lceil \gamma n \rceil$, then

$$\begin{aligned}
\sum_{r \leq i \leq n-r} \ell_{uni}(i, n-i) &= \sum_{r \leq i \leq n-r} \frac{\mathscr{C}_{i-1}\mathscr{C}_{n-i-1}}{\mathscr{C}_{n-1}} \\
&\geq \frac{(1-\varepsilon)^2}{4(1+\varepsilon)\sqrt{\pi}} \sum_{r \leq i \leq n-r} \frac{n^{3/2}}{i^{3/2}(n-i)^{3/2}} \\
&\geq \frac{(1-\varepsilon)^2 n^{3/2}}{2(1+\varepsilon)\sqrt{\pi}} \sum_{r \leq i < \frac{n}{2}} i^{-3/2}(n-i)^{-3/2} \\
&\geq \frac{(1-\varepsilon)^2 n^{3/2}}{2(1+\varepsilon)\sqrt{\pi}} \int_r^{\frac{n}{2}} x^{-3/2}(n-x)^{-3/2} \mathrm{d}x \\
&= \frac{(1-\varepsilon)^2}{(1+\varepsilon)\sqrt{\pi}} \frac{n-2r}{\sqrt{n(n-r)r}}.
\end{aligned}$$

If N_ε is chosen large enough, then we also have

$$\frac{n-2r}{\sqrt{n(n-r)}} \leq 1 - 2\gamma$$

for all r, n with $n \geq N_\varepsilon$ and $r \leq \lceil n\gamma \rceil$. So set $\delta := \frac{(1+\varepsilon)}{(1-\varepsilon)^2}$, and define

$$\varsigma_{uni}(r) := \frac{(1-2\gamma)}{\delta\sqrt{\pi r}}.$$

Then $\ell_{uni} \in \mathcal{L}_{sbal}(\varsigma_{uni}, \gamma)$. By Theorem 3.20, we thus obtain

$$\mathbb{E}(F_{n,\ell_{uni}}) \leq \frac{\delta\sqrt{2\pi}}{\sqrt{\gamma}(1-2\gamma)} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)).$$

As $0 < \gamma < 1/2$ is arbitrary, we can choose the optimal value $\gamma = 1/6$ to obtain

$$\mathbb{E}(F_{n,\ell_{uni}}) \leq 3\delta\sqrt{3\pi} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)),$$

where $\delta > 1$, as $\varepsilon > 0$. In [38, 96] it is shown that

$$\mathbb{E}(F_{n,\ell_{uni}}) = \frac{2\sqrt{2}}{\sqrt{\pi}} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)).$$

Thus, except for the leading constant, the upper bound from Theorem 3.20 yields

the correct asymptotic growth of $\mathbb{E}(F_{n,\ell_{uni}})$.

Example 3.23. For the binary search tree model ℓ_{bst} from Example 3.2, we also obtain an upper bound on $\mathbb{E}(F_{n,\ell_{bst}})$ from Theorem 3.20. Let $\gamma_{bst} = 1/4$, let $n \geq 2$, and let $1 \leq r \leq \lceil \gamma n \rceil$. Then

$$\sum_{r \leq i \leq n-r} \ell_{bst}(i, n-i) = \frac{n-2r+1}{n-1} \geq \frac{1}{2}.$$

Set $\varsigma_{bst}(r) := 1/2$. By Theorem 3.20, we find

$$\mathbb{E}(F_{n,\ell_{uni}}) \leq \frac{16n}{\log n}(1 + o(1)).$$

This coincides with the asymptotic upper bound on $\mathbb{E}(F_{n,\ell_{uni}})$ from Example 3.17 as well as Example 3.12 (except for the leading constant).

3.6 An information-theoretic lower bound

So far, we have only considered upper bounds on the expected number of distinct fringe subtrees in random trees $T_{n,\ell}$. In this section and the following section, we focus on lower bounds instead. We present some classes of leaf-centric binary tree sources for which we will be able to show asymptotic lower bounds on the number $\mathbb{E}(F_{n,\ell})$ of distinct fringe subtrees in the corresponding random trees $T_{n,\ell}$.

First, we consider a specific subclass of the class of upper-bounded leaf-centric tree sources from Definition 3.8. Let $\kappa \in (0,1)$ be a constant. With $\mathcal{L}_{up}(\kappa) \subseteq \mathcal{L}$, we denote the set of mappings $\ell \in \mathcal{L}$ that are eventually bounded above by the constant κ: that is, a mapping $\ell \in \mathcal{L}$ belongs to $\mathcal{L}_{up}(\kappa)$ if there is an integer N_ℓ such that $\ell(i, n-i) \leq \kappa$ for all $n \geq N_\ell$ and all integers $1 \leq i \leq n-1$. In the following theorem, we obtain a lower bound on $\mathbb{E}(F_{n,\ell})$ for $\ell \in \mathcal{L}_{up}(\kappa)$.

Theorem 3.24. *Let $\kappa \in (0,1)$, and let $\ell \in \mathcal{L}_{up}(\kappa)$ with integer N_ℓ. Then the number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \geq \log\left(\frac{1}{\kappa}\right) \frac{n}{2(N_\ell - 1)\log n}\,(1 + o(1))\,.$$

In order to prove Theorem 3.24, we make use of an information-theoretic argument. We make the convention that $0 \cdot \log(1/0) = 0$. Let $H(T_{n,\ell})$ denote the Shannon entropy of the random variable $T_{n,\ell}$, i.e.,

$$H(T_{n,\ell}) = \sum_{t \in \mathcal{B}_n} P_\ell(t) \log\left(1/P_\ell(t)\right).$$

We obtain the following general result.

Theorem 3.25. *Let $\ell \in \mathcal{L}$. Then the number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(F_{n,\ell}) \geq \frac{H(T_{n,\ell})}{2\log n}\,(1+o(1))\,.$$

Proof. Let $\mathfrak{f}(t)$ denote the number of distinct fringe subtrees occurring in the full binary tree $t \in \mathcal{B}$. Recall that $t \in \mathcal{B}_n$ has exactly n leaves and $n-1$ internal nodes. This implies that $\mathfrak{f}(t) \leq n$ for $t \in \mathcal{B}_n$. We first show that we need at most $2\mathfrak{f}(t)(\lfloor \log n \rfloor + 1)$ many bits in order to encode t. Recall that $\mathfrak{f}(t)$ equals the size of the minimal DAG of t, i.e., the number of nodes in the directed acyclic graph obtained from t by merging identical fringe subtrees of t. As we can uniquely retrieve t from its minimal DAG (see e.g. [75]), it suffices to show that we can encode the minimal DAG of t with at most $2\mathfrak{f}(t)(\lfloor \log n \rfloor + 1)$ many bits. Without loss of generality, assume that the nodes of the minimal DAG of t are enumerated as $1, 2, \ldots, \mathfrak{f}(t)$, where $\mathfrak{f}(t)$ is the unique leaf node of the minimal DAG of t. For $1 \leq i \leq \mathfrak{f}(t) - 1$, let l_i (resp. r_i) denote the left (resp. right) child of node k. We encode each number $1, \ldots, \mathfrak{f}(t)$ by a bit string of length exactly $\lfloor \log n \rfloor + 1$. The minimal DAG of t can be uniquely encoded as the bit string $l_1 r_1 \cdots l_{\mathfrak{f}(t)-1} r_{\mathfrak{f}(t)-1}$, which has length $2(\mathfrak{f}(t) - 1)(\lfloor \log n \rfloor + 1)$. Let $\ell \in \mathcal{L}$. Shannon's coding theorem implies

$$H(T_{n,\ell}) \leq 2(\lfloor \log n \rfloor + 1) \sum_{t \in \mathcal{B}_n} P_\ell(t)\mathfrak{f}(t) = 2(\lfloor \log n \rfloor + 1)\mathbb{E}(F_{n,\ell}).$$

This finishes the proof. □

In order to show Theorem 3.24, we make use of Theorem 3.25 together with a lower bound on the entropy $H(T_{n,\ell})$ for $\ell \in \mathcal{L}_{up}(\kappa)$. We will show the following lemma.

Lemma 3.26. *Let $\kappa \in (0,1)$, and let $\ell \in \mathcal{L}_{up}(\kappa)$ with integer $N_\ell \geq 2$. Then*

$$H(T_{n,\ell}) \geq \log\left(\frac{1}{\kappa}\right)\left(\frac{n}{N_\ell - 1} - 1\right)$$

for every $n \geq N_\ell$.

In order to prove Lemma 3.26, we need the following result.

Lemma 3.27. *Let $k \geq 1$. Every full binary tree of leafsize n contains at least $\frac{n}{k} - 1$ fringe subtrees of leafsize larger than k.*

Proof. Let t be a full binary tree of leafsize n, and let $s_1, s_2, \ldots, s_r$ be the maximal fringe subtrees of leafsize at most k (i.e., the fringe subtrees that have at most k leaves and are not contained in any other fringe subtree with that property). Every leaf of t is contained in one of these: since a leaf is itself a fringe subtree of leafsize at most k (namely leafsize 1), it must be contained in a maximal fringe subtree with that property. Moreover, $s_1, s_2, \ldots, s_r$ must be disjoint, since for

any two fringe subtrees of the same tree, either one is contained in the other, or they are disjoint. Hence we have $n = \|t\| = \|s_1\| + \|s_2\| + \cdots + \|s_r\| \leq kr$, which implies $r \geq \frac{n}{k}$. Lastly, note that the roots of $s_1, s_2, \ldots, s_r$ and all internal nodes of t that are not contained in these r fringe subtrees form a full binary tree of leafsize r. This tree has $r - 1$ internal nodes, and all of them are roots of fringe subtrees of t whose leafsize is at least $k + 1$. □

In particular, we obtain the following corollary from Lemma 3.27:

Corollary 3.28. *Let $\ell \in \mathcal{L}$, and let $n > k$. The random number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,k}) \geq \frac{n}{k} - 1.$$

We are now able to prove Lemma 3.26:

Proof of Lemma 3.26. Lemma 3.26 follows from identity (4) in [66]: Define

$$h_k(\ell) = \sum_{i=1}^{k-1} \ell(i, k-i) \log\left(\frac{1}{\ell(i, k-i)}\right),$$

and recall that by convention, $0 \cdot \log(1/0) = 0$, so that this is well-defined. Thus, $h_k(\ell)$ is the Shannon entropy of the random variable corresponding to the probability mass function $\ell\colon \{(i, k-i) \mid 1 \leq i \leq k-1\} \to [0, 1]$. As $\ell(i, k-i) \leq \kappa$ for $k \geq N_\ell$, we find

$$h_k(\ell) \geq \log(1/\kappa) \sum_{i=1}^{k-1} \ell(i, k-i) = \log(1/\kappa)$$

for every $k \geq N_\ell$. Identity (4) in [66] states that

$$H(T_{n,\ell}) = \sum_{k=2}^{n} \left(\mathbb{E}(Y_{n,k-1}) - \mathbb{E}(Y_{n,k})\right) h_k(\ell),$$

where $Y_{n,k}$ again denotes the random number of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$. For every $n \geq N_\ell \geq 2$, we obtain

$$\begin{aligned}
H(T_{n,\ell}) &\geq \sum_{k=N_\ell}^{n} \left(\mathbb{E}(Y_{n,k-1}) - \mathbb{E}(Y_{n,k})\right) h_k(\ell) \\
&\geq \log\left(\frac{1}{\kappa}\right) \sum_{k=N_\ell}^{n} \left(\mathbb{E}(Y_{n,k-1}) - \mathbb{E}(Y_{n,k})\right) \\
&= \log\left(\frac{1}{\kappa}\right) \left(\mathbb{E}(Y_{n,N_\ell-1}) - \mathbb{E}(Y_{n,n})\right) \\
&= \log\left(\frac{1}{\kappa}\right) \mathbb{E}(Y_{n,N_\ell-1}).
\end{aligned}$$

By Corollary 3.28, this implies

$$H(T_{n,\ell}) \geq \log\left(\frac{1}{\kappa}\right)\left(\frac{n}{N_\ell - 1} - 1\right).$$

This proves the statement. □

Theorem 3.24 now follows immediately from Theorem 3.25 and Lemma 3.26. From Theorem 3.24, we obtain the following lower bound on the number of distinct fringe subtrees in a random binary search tree.

Example 3.29. Let us apply Theorem 3.25 to the binary search tree model from Example 3.2. In [66], it is shown that

$$H(T_{n,\ell_{bst}}) \sim 1.7363771368\, n\,(1 + o(1)).$$

Hence, we obtain from Theorem 3.25 that

$$\mathbb{E}(F_{n,\ell_{bst}}) \geq \frac{0.8681\, n}{\log n}(1 + o(1)). \tag{3.16}$$

In Example 3.12, we have shown the upper bound

$$\mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{8n}{\log n}(1 + o(1))$$

(see also Examples 3.17 and 3.23). We thus find $\mathbb{E}(F_{n,\ell_{bst}}) = \Theta(n/\log n)$. Recall again that Devroye already proved in [24], that

$$\log(3)/2 \cdot \frac{n}{\log n}(1 + o(1)) \leq \mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{4n}{\log n}(1 + o(1)),$$

where the upper bound was also shown in [36]. In particular, the lower bound (3.16) improves the lower bound from [24]. The upper bounds from Example 3.12, Example 3.17 and Example 3.23 are weaker in terms of the leading constant than the upper bounds shown in [36, 24], but however, they are derived from much more general results that cover whole classes of distributions. We remark again that in Chapter 5 in Corollary 5.6, we show that

$$c_1 \cdot \frac{n}{\log n}(1 + o(1)) \leq \mathbb{E}(F_{n,\ell_{bst}}) \leq \frac{4n}{\log n}(1 + o(1)),$$

where $c_1 \approx 3.472754274$.

Similarly, we can apply Theorem 3.24 in order to obtain a lower bound on the number of distinct fringe subtrees in a random tree generated by the binomial random tree model. Specifically, the next example shows that the expected number of distinct fringe subtrees in a random tree of size n generated by the binomial random tree model is in $\Omega(n/\log n)$. Together with the upper bound (Example 3.18), we hence obtain $\mathbb{E}(F_{n,\ell_{bin,p}}) = \Theta(n/\log n)$.

Example 3.30. For the binomial random tree model (Example 3.4), a simple induction over n (using $\binom{n-2}{i-1} = \binom{n-3}{i-1} + \binom{n-3}{i-2}$ for $2 \leq i \leq n-2$ in the induction step) shows that for $n \geq 3$, we have

$$\ell_{bin,p}(i, n-i) \leq \max\{p, 1-p\}$$

for all $1 \leq i \leq n-1$. With $N_\ell = 3$ and $\kappa = \max\{p, 1-p\}$, we thus obtain from Theorem 3.24 that

$$\mathbb{E}(F_{n,\ell_{bin,p}}) \geq \log\left(\frac{1}{\kappa}\right) \frac{n}{4\log n}(1 + o(1)). \tag{3.17}$$

If we consider the proof of Lemma 3.26 again, we can slightly improve the leading constant in the lower bound (3.17). With identity (4) in [66], $h_2(\ell) = 0$, and Corollary 3.28, we get

$$\begin{aligned} H(T_{n,\ell}) &= \sum_{k=2}^{n} \left(\mathbb{E}(Y_{n,k-1}) - \mathbb{E}(Y_{n,k})\right) h_k(\ell) \\ &\geq \min_{3 \leq i \leq n} h_i(\ell) \cdot \sum_{k=3}^{n} \left(\mathbb{E}(Y_{n,k-1}) - \mathbb{E}(Y_{n,k})\right) \\ &= \min_{3 \leq i \leq n} h_i(\ell) \cdot \mathbb{E}(Y_{n,2}) \\ &\geq \min_{3 \leq i \leq n} h_i(\ell) \cdot \left(\frac{n}{2} - 1\right). \end{aligned}$$

Together with Theorem 3.25, we thus obtain

$$\mathbb{E}(F_{n,\ell}) \geq \min_{3 \leq i \leq n} h_i(\ell) \cdot \frac{n}{4\log n}(1 + o(1)).$$

For the binomial random tree model we have $h_k(\ell_{bin,p}) \leq h_{k+1}(\ell_{bin,p})$: if $\mathrm{Ber}(p)$ is a Bernoulli random variable that takes the value 1 (resp., 0) with probability p (resp., $1-p$), then the binomial random variable $\mathrm{Bin}(k,p)$ is the sum of k independent copies of $\mathrm{Ber}(p)$. Hence, we have

$$\mathrm{Bin}(k+1, p) = \mathrm{Bin}(k, p) + \mathrm{Ber}(p).$$

Moreover, for independent random variables X and Y the Shannon entropy satisfies $H(X+Y) \geq H(X)$; see e.g. [78]. Hence, we get

$$\min_{3 \leq i \leq n} h_i(\ell_{bin,p}) = h_3(\ell_{bin,p}) = p\log\left(\frac{1}{p}\right) + (1-p)\log\left(\frac{1}{1-p}\right).$$

This yields

$$\mathbb{E}(F_{n,\ell_{bin,p}}) \geq \left(p\log\left(\frac{1}{p}\right) + (1-p)\log\left(\frac{1}{1-p}\right)\right) \cdot \frac{n}{4\log n}(1 + o(1)).$$

Together with the upper bound (Example 3.18), we get $\mathbb{E}(F_{n,\ell_{bin,p}}) = \Theta(n/\log n)$.

Example 3.31. For the uniform distribution from Example 3.3, we have [66]

$$H(T_{n,\ell_{uni}}) \sim 2n(1 + o(1)).$$

Theorem 3.25 thus yields a non-trivial lower bound

$$\mathbb{E}(F_{n,\ell_{uni}}) \geq \frac{n}{\log n}(1 + o(1)).$$

However, recall that in [38], it is shown that

$$\mathbb{E}(F_{n,\ell_{uni}}) = \frac{2\sqrt{2}}{\sqrt{\pi}} \cdot \frac{n}{\sqrt{\log n}} (1 + o(1)) \tag{3.18}$$

(see also Corollary 4.18 in Chapter 4, where it is shown that the estimate (3.18) holds not only in expectation, but also with high probability, that is, with probability tending to 1 as $n \to \infty$). In the next section, we will prove a general result (for a whole class of leaf-centric tree sources) that implies a lower bound of the form $\mathbb{E}(F_{n,\ell_{uni}}) \geq \Omega\left(n/\sqrt{\log n}\right)$ (see Example 3.37).

3.7 Unbalanced sources

For upper-bounded sources, Theorem 3.24 can only yield lower bounds of the form $\Omega(n/\log n)$ for $\mathbb{E}(F_{n,\ell})$. In this section, we present another class of leaf-centric binary tree sources, for which we will be able to derive stronger lower bounds on $\mathbb{E}(F_{n,\ell})$, such as the lower bound $\Omega(n/\sqrt{\log n})$ for the uniform distribution.

Definition 3.32 (ϱ-unbalanced sources)**.** Let $\varrho\colon \mathbb{R} \to (0, 1]$ be a decreasing function, and let $\gamma \in (0, \frac{1}{2})$. With $\mathcal{L}_{unbal}(\varrho, \gamma) \subseteq \mathcal{L}$, we denote the set of mappings ℓ for which there are integers N'_ℓ and c_ℓ such that

$$\sum_{r \leq i \leq n-r} \ell(i, n-i) \leq \varrho(r)$$

for every $n \geq N'_\ell$ and every $c_\ell \leq r \leq \lceil \gamma n \rceil$.

Note that every leaf-centric binary tree source is unbalanced with respect to the constant function ϱ with $\varrho(x) = 1$ for every $x \in \mathbb{R}$. Moreover, let $\kappa \in (0, 1)$ be a constant. As in the previous section, let again $\mathcal{L}_{up}(\kappa) \subseteq \mathcal{L}$ denote the set of mappings $\ell \in \mathcal{L}$ that are upper-bounded with respect to the constant κ. Our main result in this section is the following theorem:

Theorem 3.33. *Let ϱ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$ and $\kappa \in (0, 1)$. Moreover, let $\ell \in \mathcal{L}_{up}(\kappa)$ with integer N_ℓ, and let $\ell \in \mathcal{L}_{unbal}(\varrho, \gamma)$ with integer N'_ℓ and constant c_ℓ. If*

(a) $\varrho(x) \geq \omega(1/x)$,

(b) there is a constant $c_\varrho > 0$ such that $1/c_\varrho + \gamma \leq 1$ and

$$\sum_{i=c_\ell}^{k} \varrho(i) \leq c_\varrho(k+1)\varrho(k+1)$$

for every $k \geq c_\ell$, and

(c) there is an integer $j_\ell \geq 2$ such that the number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies

$$\mathbb{E}(Y_{n,(\log n)^{j_\ell}}) \leq o(n/\log n),$$

then the number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies

$$\mathbb{E}(F_{n,\ell}) \geq \frac{\gamma n \log(1/\kappa)}{c_\varrho N_\ell \varrho\big(\frac{N_\ell \log n}{\log(1/\kappa)}\big) \log n}\,(1+o(1))\,.$$

As every leaf-centric binary tree source is unbalanced with respect to the constant function ϱ with $\varrho(x) = 1$ for every $x \in \mathbb{R}$, we obtain the following corollary of Theorem 3.33, which is a weaker version of Theorem 3.24:

Corollary 3.34. *Let $\kappa \in (0,1)$, and let $\ell \in \mathcal{L}_{up}(\kappa)$ with integer N_ℓ. If there is an integer $j_\ell \geq 2$ such that the number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies*

$$\mathbb{E}(Y_{n,(\log n)^{j_\ell}}) \leq o(n/\log n),$$

then the number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ satisfies

$$\mathbb{E}(F_{n,\ell}) \geq \frac{\gamma n \log(1/\kappa)}{2N_\ell \log n}(1+o(1)),$$

for every $\gamma \in (0, \frac{1}{2})$.

Proof. Let $\ell \in \mathcal{L}$ satisfy the requirements of Corollary 3.34. We show that ℓ then satisfies all further requirements of Theorem 3.33 as well. Clearly, ℓ is ϱ_1-unbalanced for every constant $\gamma \in (0, 1/2)$, where ϱ_1 is the constant function defined by $\varrho_1(x) = 1$ for every $x \in \mathbb{R}$. Hence, we have $\ell \in \mathcal{L}_{unbal}(\varrho_1, \gamma)$ with $N'_\ell = 1$ and $c_\ell = 1$. We now find that condition (a) of Theorem 3.33 is clearly satisfied, as is condition (b), where the constant c_ϱ can be chosen for example as $c_\varrho = 2$. Finally, condition (c) of Theorem 3.33 is satisfied by assumption. Thus, we obtain the lower bound

$$\mathbb{E}(F_{n,\ell}) \geq \frac{\gamma n \log(1/\kappa)}{2N_\ell \log n}(1+o(1))$$

from Theorem 3.33, where $\gamma \in (0, 1/2)$ is arbitrary. □

Note that Corollary 3.34 resembles Theorem 3.24 in many ways, except that we have the additional requirement (c) from Theorem 3.33 and the additional

constant $\gamma \in (0, 1/2)$ in the lower bound. The technique to prove Theorem 3.33 however will be quite different from the technique used for Theorem 3.24.

In order to prove Theorem 3.33, we make use of a refinement of the cut-point technique from Lemma 3.5, as was applied to a similar problem in [96]. Furthermore, the very same refinement of the cut-point argument will be used in Chapter 4 and Chapter 5 of this work in order to obtain lower bounds on the average number of fringe subtrees. More precisely, we refine the cut-point technique with an inclusion-exclusion-principle-like argument in order to obtain a lower bound on the expected number of distinct fringe subtrees. We first need a lower bound on the expected number $\mathbb{E}(Y_{n,k})$ of fringe subtrees of leafsize larger than k in a random tree $T_{n,\ell}$, where $\ell \in \mathcal{L}_{unbal}(\varrho, \gamma)$:

Lemma 3.35. *Let $\varrho\colon \mathbb{R} \to (0, 1]$ be a decreasing function, let $\gamma \in (0, \frac{1}{2})$, and let $\ell \in \mathcal{L}_{unbal}(\varrho, \gamma)$ with integers N'_ℓ and c_ℓ. If*

(a) $\varrho(x) \geq \omega(1/x)$, and

(b) there is a constant $c_\varrho > 0$ such that $1/c_\varrho + \gamma \leq 1$ and for every $k \geq c_\ell$

$$\sum_{i=c_\ell}^{k} \varrho(i) \leq c_\varrho(k+1)\varrho(k+1),$$

then there is an integer m such that for $k \geq m$ and $n \geq k+1$, the random number $Y_{n,k}$ of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$ satisfies

$$\mathbb{E}(Y_{n,k}) \geq \frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)}.$$

In order to prove Lemma 3.35, the following lemma will be helpful.

Lemma 3.36 (Summation by parts, [99, Theorem. 3.41])**.** *Let m be a natural number and $a_1, \ldots, a_m$, $b_1, \ldots, b_m$ be real numbers. Then*

$$\sum_{i=1}^{m} a_i b_i = \left(\sum_{i=1}^{m} a_i\right) b_m + \sum_{i=1}^{m-1} \left(\sum_{\ell=1}^{i} a_\ell\right) (b_i - b_{i+1}).$$

We are now able to prove Lemma 3.35.

Proof of Lemma 3.35. As $\varrho(x) \geq \omega(1/x)$ by condition (a), there is an integer N_ϱ such that

$$\frac{c_\ell - 1}{\varrho(k+1)} \leq k \tag{3.19}$$

for all $k \geq N_\varrho$. We choose the integer m in such a way that $m \geq \max\{N_\varrho, N'_\ell, c_\ell\}$ and $\lceil \gamma n \rceil \leq n/2$ for $n > m$. We prove the statement using induction for $n \geq k+1 > m$. For the base case, let $n = k+1$. A full binary tree $t \in \mathcal{B}_{k+1}$ has

exactly one fringe subtree of leafsize larger than k, and we have

$$\mathbb{E}(Y_{k+1,k}) = 1 > \frac{\gamma}{c_\varrho \varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)}.$$

For the induction step, take an integer $n > k+1$, so that

$$\mathbb{E}(Y_{i,k}) \geq \frac{\gamma i}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)}$$

for every integer $k+1 \leq i \leq n-1$ by induction hypothesis. We distinguish two cases in the induction step.

Case 1: $\lceil \gamma n \rceil < k+1 \leq n-1$. We thus have

$$\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)} < \frac{1}{c_\varrho \varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)} = 0.$$

Furthermore, as $n > k$, we find $\mathbb{E}(Y_{n,k}) \geq 1$. The statement follows in this case.

Case 2: $k+1 \leq \lceil \gamma n \rceil$. As $\lceil \gamma n \rceil \leq n/2$ for $n > m$, we find by Lemma 3.6:

$$\mathbb{E}(Y_{n,k}) = 1 + \sum_{i=k+1}^{n-k-1} \ell(i, n-i) \left(\mathbb{E}(Y_{i,k}) + \mathbb{E}(Y_{n-i,k})\right) + \sum_{i=n-k}^{n-1} \ell^*(i, n-i)\mathbb{E}(Y_{i,k}).$$

By the induction hypothesis, we have

$$\begin{aligned}\mathbb{E}(Y_{n,k}) \geq 1 &+ \sum_{i=k+1}^{n-k-1} \ell(i, n-i) \left(\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{2}{c_\varrho \varrho(k+1)}\right) \\ &+ \sum_{i=n-k}^{n-1} \ell^*(i, n-i) \left(\frac{\gamma i}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)}\right).\end{aligned}$$

We introduce the abbreviation

$$S(i) = \frac{\gamma i}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho \varrho(k+1)}.$$

Clearly, $S(i)$ is increasing in i. Moreover, set

$$\sum_{i=k+1}^{n-k-1} \ell(i, n-i) =: \alpha_1, \sum_{i=n-k}^{n-c_\ell} \ell^*(i, n-i) =: \alpha_2 \text{ and } \sum_{i=n-c_\ell+1}^{n-1} \ell^*(i, n-i) =: \alpha_3.$$

We obtain

$$\begin{aligned}\mathbb{E}(Y_{n,k}) \geq 1 &+ \alpha_1 \left(\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{2}{c_\varrho \varrho(k+1)}\right) + \alpha_3 S(n - c_\ell + 1) \\ &+ \sum_{i=n-k}^{n-c_\ell} \ell^*(i, n-i) S(i). \end{aligned} \tag{3.20}$$

As $\ell \in \mathcal{L}_{unbal}(\varrho,\gamma)$ and additionally $N'_\ell < n$ and $c_\ell < k+1 \le \lceil \gamma n \rceil$, we have $0 \le \alpha_1 \le \varrho(k+1)$, $1-\varrho(c_\ell) \le \alpha_3 \le 1$ and $\alpha_1+\alpha_2+\alpha_3 = 1$. Furthermore, we have

$$\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{2}{c_\varrho \varrho(k+1)} \le S(i)$$

for every integer i with $n-k \le i \le n-c_\ell+1$. Hence, we minimize the right-hand side of the identity (3.20) subject to the condition $\alpha_1+\alpha_2+\alpha_3 = 1$ if we assign the maximal possible weight $\alpha_1 = \varrho(k+1)$ to the smallest term $\gamma n/(c_\varrho(k+1)\varrho(k+1)) - 2/(c_\varrho\varrho(k+1))$ and the minimal weight $\alpha_3 = 1-\varrho(c_\ell)$ to the largest term $S(n-c_\ell+1)$. It remains to bound the sum $\sum_{i=n-k}^{n-c_\ell} \ell^*(i,n-i)S(i)$ subject to the condition $\alpha_2 = \varrho(c_\ell) - \varrho(k+1)$. First note that $\ell \in \mathcal{L}_{unbal}(\varrho,\gamma)$, $N'_\ell < n$ and $\alpha_3 = 1-\varrho(c_\ell)$ imply

$$\sum_{j=c_\ell}^{i} \ell^*(j,n-j) = \sum_{j=1}^{i} \ell^*(j,n-j) - \alpha_3 \ge \varrho(c_\ell) - \varrho(i+1), \tag{3.21}$$

for every $c_\ell \le i \le k$ (note that $c_\ell \le i+1 \le \lceil \gamma n \rceil$; therefore we can take $r = i+1$ in Definition 3.32). Using summation by parts (Lemma 3.36), we obtain

$$\begin{aligned}
&\sum_{i=n-k}^{n-c_\ell} \ell^*(i,n-i)S(i) = \sum_{i=c_\ell}^{k} \ell^*(i,n-i)S(n-i) \\
&= S(n-k)\sum_{i=c_\ell}^{k} \ell^*(i,n-i) + \sum_{i=c_\ell}^{k-1} \Big(\sum_{j=c_\ell}^{i} \ell^*(j,n-j) \Big) \left(S(n-i) - S(n-i-1) \right) \\
&\ge \left(\varrho(c_\ell) - \varrho(k+1) \right) S(n-k) + \sum_{i=c_\ell}^{k-1} (\varrho(c_\ell) - \varrho(i+1))(S(n-i) - S(n-i-1)) \\
&= -\varrho(k+1)S(n-k) + \varrho(c_\ell)S(n-c_\ell) - \sum_{i=c_\ell}^{k-1} \varrho(i+1) \left(S(n-i) - S(n-i-1) \right)
\end{aligned}$$

where the inequality follows from (3.21). Note that we have

$$S(n-i) - S(n-i-1) = \frac{\gamma}{c_\varrho(k+1)\varrho(k+1)}.$$

Altogether, we obtain from (3.20) that

$$\begin{aligned}
\mathbb{E}(Y_{n,k}) \ge 1 &+ \varrho(k+1)\left(\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{2}{c_\varrho\varrho(k+1)} \right) - \sum_{i=c_\ell}^{k} \frac{\varrho(i)\gamma}{c_\varrho(k+1)\varrho(k+1)} \\
&+ \frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{\gamma(c_\ell-1)}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho\varrho(k+1)} \\
&- \varrho(k+1)\left(\frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{\gamma k}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho\varrho(k+1)} \right).
\end{aligned}$$

Simplifying the right-hand side yields

$$\mathbb{E}(Y_{n,k}) \geq 1 - \frac{1}{c_\varrho} + \frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{\gamma(c_\ell - 1)}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho\varrho(k+1)} + \frac{\gamma k}{c_\varrho(k+1)} - \sum_{i=c_\ell}^{k} \frac{\varrho(i)\gamma}{c_\varrho(k+1)\varrho(k+1)}.$$

Since we have

$$\sum_{i=c_\ell}^{k} \varrho(i) \leq c_\varrho(k+1)\varrho(k+1)$$

by the assumption from condition (b), we get

$$\mathbb{E}(Y_{n,k}) \geq 1 - \frac{1}{c_\varrho} + \frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{\gamma(c_\ell - 1)}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho\varrho(k+1)} + \frac{\gamma k}{c_\varrho(k+1)} - \gamma.$$

Furthermore, from (3.19) we obtain

$$-\frac{\gamma(c_\ell - 1)}{c_\varrho(k+1)\varrho(k+1)} + \frac{\gamma k}{c_\varrho(k+1)} \geq 0$$

and hence

$$\mathbb{E}(Y_{n,k}) \geq 1 - \frac{1}{c_\varrho} + \frac{\gamma n}{c_\varrho(k+1)\varrho(k+1)} - \frac{1}{c_\varrho\varrho(k+1)} - \gamma.$$

Finally, as $1/c_\varrho + \gamma \leq 1$ by condition (b), the statement follows. □

We are now able to prove Theorem 3.33. The main idea of the proof is based on techniques from [96] (see also the proofs of Theorem 4.13 and Theorem 5.3 in Chapter 4 and Chapter 5).

Proof of Theorem 3.33. Let δ_n for $n \geq 2$ be defined by $\delta_n = 2j_\ell \log\log n / \log n$, where $j_\ell \geq 2$ is the constant from condition (c) in Theorem 3.33. Moreover, let

$$k_1 = \left\lceil N_\ell\left(1 + \frac{(1+\delta_n)\log n}{\log(1/\kappa)}\right)\right\rceil.$$

Furthermore, assume that n is sufficiently large, so that the lower bound on $\mathbb{E}(Y_{n,k})$ from Lemma 3.35 applies for $k \geq k_1 - 1$, and that $k_1 < (\log n)^{j_\ell}$. The random number $F_{n,\ell}$ of distinct fringe subtrees in the random tree $T_{n,\ell}$ is bounded below by the number of distinct fringe subtrees in $T_{n,\ell}$ of leafsizes k for $k_1 \leq k \leq n^{\delta_n/2}$, where $n^{\delta_n/2} = (\log n)^{j_\ell}$ by definition of δ_n. Let $X_{n,k}$ denote the (random) number of fringe subtrees of leafsize exactly k occurring in $T_{n,\ell}$, and let $X_{n,k}^{(2)}$ denote the (random) number of unordered pairs of identical fringe subtrees of leafsize exactly k occurring in a random tree $T_{n,\ell}$. Using the

inclusion-exclusion principle, we find that $\mathbb{E}(F_{n,\ell})$ can be bounded from below as follows:

$$\mathbb{E}(F_{n,\ell}) \geq \sum_{k_1 \leq k \leq n^{\delta_n/2}} \mathbb{E}(X_{n,k}) - \sum_{k_1 \leq k \leq n^{\delta_n/2}} \mathbb{E}(X_{n,k}^{(2)}). \tag{3.22}$$

We first bound the second sum on the right-hand side of (3.22). By Lemma 3.27, we find that the number of fringe subtrees of leafsize at least $N_\ell + 1$ in a full binary tree t of leafsize $k > N_\ell$ is at least $k/N_\ell - 1$. As $\ell \in \mathcal{L}_{up}(\kappa)$ with integer N_ℓ, we find

$$P_\ell(t) = \prod_{v \in V_0(t)} \ell(\|t_l[v]\|, \|t_r[v]\|) \leq \prod_{\substack{v \in V(t) \\ \|t[v]\| > N_\ell}} \ell(\|t_l[v]\|, \|t_r[v]\|) \leq \kappa^{k/N_\ell - 1}$$

for every full binary tree $t \in \mathcal{B}_k$. Let us condition on the event that $X_{n,k} = N$ for some natural number N. These N fringe subtrees are independent random trees of leafsize k, and the probability that such a fringe subtree equals a given full binary tree $t \in \mathcal{B}_k$ is given by $P_\ell(t)$. Thus, we have

$$\mathbb{E}(X_{n,k}^{(2)} \mid X_{n,k} = N) = \binom{N}{2} \sum_{t \in \mathcal{B}_k} P_\ell(t)^2 \leq n^2 \kappa^{k/N_\ell - 1} \sum_{t \in \mathcal{B}_k} P_\ell(t) = n^2 \kappa^{k/N_\ell - 1}.$$

Since this upper bound on $\mathbb{E}(X_{n,k}^{(2)} \mid X_{n,k} = N)$ holds independently of N, we obtain with the law of total expectation

$$\mathbb{E}(X_{n,k}^{(2)}) = \sum_{N=0}^{n} \mathbb{E}(X_{n,k}^{(2)} \mid X_{n,k} = N) \mathbb{P}(X_{n,k} = N) \leq n^2 \kappa^{k/N_\ell - 1}.$$

Since $k_1 \geq N_\ell \big(1 + \frac{(1+\delta_n)\log n}{\log(1/\kappa)}\big)$, we obtain for all $k \geq k_1$:

$$\mathbb{E}(X_{n,k}^{(2)}) \leq n^{1-\delta_n}.$$

We thus have

$$\sum_{k_1 \leq k \leq n^{\delta_n/2}} \mathbb{E}(X_{n,k}^{(2)}) \leq n^{1-\delta_n/2} = \frac{n}{(\log n)^{j_\ell}} \leq o\left(\frac{n}{\log n}\right).$$

It remains to bound the first sum on the right-hand side of (3.22): If $Y_{n,k}$ again denotes the random number of fringe subtrees of leafsize larger than k in the random tree $T_{n,\ell}$, we have

$$\sum_{k_1 \leq k \leq n^{\delta_n/2}} \mathbb{E}(X_{n,k}) = \mathbb{E}(Y_{n,k_1-1}) - \mathbb{E}(Y_{n,n^{\delta_n/2}}).$$

By condition (c) of Theorem 3.33, we have $\mathbb{E}(Y_{n,n^{\delta_n/2}}) \leq o(n/\log n)$. Moreover,

by Lemma 3.35, we find that

$$\mathbb{E}(Y_{n,k_1-1}) \geq \frac{\gamma n}{c_\varrho k_1 \varrho(k_1)} - \frac{1}{c_\varrho \varrho(k_1)} \geq \Omega\left(\frac{n}{\log n}\right),$$

since $\varrho(x) \leq 1$. By monotonicity of ϱ, we have $\varrho(k_1) \leq \varrho\big(\frac{N_\ell \log n}{\log(1/\kappa)}\big)$, so we obtain from (3.22) that

$$\mathbb{E}(F_{n,\ell}) \geq \frac{\gamma \log(1/\kappa) n}{c_\varrho N_\ell \varrho\big(\frac{N_\ell \log n}{\log(1/\kappa)}\big) \log n} \left(1 + o(1)\right).$$

This finishes the proof. □

For the uniform model from Example 3.3, we obtain the following lower bound from Theorem 3.33.

Example 3.37. In this example, we use Theorem 3.33 in order to prove a lower bound on the expected number of distinct fringe subtrees in a random tree $T_{n,\ell_{uni}}$, i.e., in a random full binary tree of leafsize n drawn from the set $\mathcal{B}_n$ according to the uniform probability distribution ℓ_{uni} from Example 3.3. We have to show that all requirements of Theorem 3.33 are satisfied. Let us write ℓ for ℓ_{uni} in the following. Recall from Example 3.3 that

$$\ell(i, n-i) = \frac{\mathscr{C}_{i-1}\mathscr{C}_{n-i-1}}{\mathscr{C}_{n-1}}.$$

(i) $\ell \in \mathcal{L}_{up}(\kappa)$ for a constant $\kappa \in (0,1)$. A short computation shows that for a fixed integer n, the maximal value of $\ell(i, n-i)$ with $1 \leq i \leq n-1$ is attained at $i_{max}(n) = 1$ and $i'_{max}(n) = n-1$. In particular, we have

$$\ell(i, n-i) \leq \ell(1, n-1) \leq \frac{n}{4n-6}$$

for all $1 \leq i \leq n-1$. If we fix N_ℓ, we thus find that $\ell \in \mathcal{L}_{up}(\kappa)$ with integer N_ℓ, where $\kappa = N_\ell/(4N_\ell - 6)$. For the sake of simplicity, let $N_\ell = 3$. Then $\ell \in \mathcal{L}_{up}(\kappa)$ with $\kappa = 1/2$.

(ii) $\ell \in \mathcal{L}_{unbal}(\varrho, \gamma)$ for a decreasing function ϱ and a constant γ. Let $0 < \gamma < 1/2$, and let $\varepsilon_0 > 0$ and $\delta_0 > 1$. From the asymptotic formula for the Catalan numbers (2.1), we find that there is an integer N'_ℓ (depending on ε_0 and δ_0) such that both

$$\frac{(1-\varepsilon_0)4^n}{\sqrt{\pi} n^{3/2}} \leq \mathscr{C}_n \leq \frac{(1+\varepsilon_0)4^n}{\sqrt{\pi} n^{3/2}}$$

and

$$\frac{n^{5/2}}{(n-2)^2\sqrt{(r-2)(n-\gamma n-1)}} \leq \frac{\delta_0}{\sqrt{(1-\gamma)r}}$$

for all $n, r \geq N'_\ell$. Set $c_\ell = N'_\ell$, and suppose that $c_\ell \leq r \leq \lceil \gamma n \rceil$. Then

$$\begin{aligned}
\sum_{r \leq i \leq n-r} \ell(i, n-i) &= \sum_{r \leq i \leq n-r} \frac{\mathscr{C}_{i-1}\mathscr{C}_{n-i-1}}{\mathscr{C}_{n-1}} \\
&\leq \frac{(1+\varepsilon_0)^2 (n-1)^{3/2}}{(1-\varepsilon_0) 4\sqrt{\pi}} \sum_{r \leq i \leq n-r} (i-1)^{-3/2} (n-i-1)^{-3/2} \\
&\leq \frac{(1+\varepsilon_0)^2 n^{3/2}}{(1-\varepsilon_0) 2\sqrt{\pi}} \int_{r-1}^{n/2} (x-1)^{-3/2} (n-x-1)^{-3/2} \mathrm{d}x \\
&= \frac{(1+\varepsilon_0)^2 n^{3/2}}{(1-\varepsilon_0)\sqrt{\pi}} \frac{n-2r+2}{(n-2)^2 \sqrt{(r-2)(n-r)}} \\
&\leq \frac{(1+\varepsilon_0)^2}{(1-\varepsilon_0)\sqrt{\pi}} \frac{n^{5/2}}{(n-2)^2 \sqrt{(r-2)(n-\gamma n-1)}} \\
&\leq \frac{(1+\varepsilon_0)^2}{(1-\varepsilon_0)\sqrt{\pi}} \frac{\delta_0}{\sqrt{(1-\gamma) r}}.
\end{aligned}$$

We thus find that $\ell \in \mathcal{L}_{unbal}(\varrho, \gamma)$ if we set

$$\delta_1 := \frac{(1+\varepsilon_0)^2 \delta_0}{1-\varepsilon_0} > 1 \quad \text{and} \quad \varrho(r) := \frac{\delta_1}{\sqrt{\pi(1-\gamma) r}},$$

where the constant $\delta_1 > 1$ can take values arbitrarily close to 1 for suitable choices of ε_0 and δ_0.

(iii) Condition (a) of Theorem 3.33 is clearly satisfied, i.e., $\varrho(x) \geq \omega(1/x)$. Condition (b) is satisfied as well: We have

$$\begin{aligned}
\sum_{i=N'_\ell}^{k} \varrho(i) &= \frac{\delta_1}{\sqrt{\pi(1-\gamma)}} \sum_{i=N'_\ell}^{k} i^{-1/2} \leq \frac{\delta_1}{\sqrt{\pi(1-\gamma)}} \int_{N'_\ell - 1}^{k} x^{-1/2} \mathrm{d}x \\
&\leq \frac{2\delta_1 \sqrt{k+1}}{\sqrt{\pi(1-\gamma)}} = 2(k+1)\varrho(k+1).
\end{aligned}$$

Thus, condition (b) holds with $c_\varrho = 2$ (as $\gamma < 1/2$, we have $1/c_\varrho + \gamma \leq 1$ as well).

(iv) Finally, condition (c) of Theorem 3.33 holds as well: From Example 3.22, we know that $\ell \in \mathcal{L}_{sbal}(\varsigma, \gamma)$ with $\varsigma(x) = \Theta(x^{-1/2})$ for $0 < \gamma < 1/2$. If we set $j_\ell = 4$, then Lemma 3.21 yields

$$\mathbb{E}(Y_{n,(\log n)^4}) \leq o\left(\frac{n}{\log n}\right).$$

We can therefore apply Theorem 3.33. The expected number of distinct fringe subtrees in a random tree of leafsize n drawn according to the uniform probability

distribution satisfies

$$\begin{aligned}\mathbb{E}(F_{n,\ell_{uni}}) &\geq \frac{\gamma n \log(1/\kappa)}{c_\varrho N_\ell \varrho\big(\frac{N_\ell \log n}{\log(1/\kappa)}\big) \log n}(1+o(1)) \\ &= \frac{\gamma\sqrt{(1-\gamma)\pi\log(1/\kappa)}}{\delta_1 c_\varrho \sqrt{N_\ell}} \cdot \frac{n}{\sqrt{\log n}}(1+o(1)) \\ &= \frac{\gamma\sqrt{(1-\gamma)\pi}}{2\sqrt{3}\delta_1} \cdot \frac{n}{\sqrt{\log n}}(1+o(1)).\end{aligned}$$

As $0 < \gamma < 1/2$ is arbitrary, we can choose γ arbitrarily close to the optimal value $1/2$. We then obtain for $\ell = \ell_{uni}$

$$\mathbb{E}(F_{n,\ell_{uni}}) \geq \frac{\sqrt{\pi}}{4\sqrt{6}\delta_1} \cdot \frac{n}{\sqrt{\log n}}(1+o(1))$$

for any constant $\delta_1 > 1$. Recall that in Example 3.22, we have already shown that

$$\mathbb{E}(F_{n,\ell_{uni}}) \leq 3\delta\sqrt{3\pi} \cdot \frac{n}{\sqrt{\log n}}(1+o(1)),$$

for any constant $\delta > 1$. In particular, we thus have $\mathbb{E}(F_{n,\ell_{uni}}) = \Theta(n/\sqrt{\log n})$. We remark again that in [38] (see also Corollary 4.18), it is shown that in fact

$$\mathbb{E}(F_{n,\ell_{uni}}) = \frac{2\sqrt{2}}{\sqrt{\pi}} \cdot \frac{n}{\sqrt{\log(n)}}(1+o(1)).$$

3.8 Conclusion and open problems

In this chapter, we proposed several classes of leaf-centric binary tree sources and derived upper and lower bounds on the number of distinct fringe subtrees occurring in a random full binary tree generated by a leaf-centric binary tree source from the respective class.

Another type of binary tree sources are *depth-centric binary tree sources* [S3], [105], which yield probability distributions on the set of full binary trees of a fixed depth and resemble leaf-centric tree sources in many ways. Furthermore, leaf-centric binary tree sources are generalized to random tree models for plane trees, which are called *fixed-size ordinal tree sources* in [S7].

An interesting problem would be to estimate the number of distinct fringe subtrees with respect to classes of depth-centric or fixed-size ordinal tree sources. Furthermore, another open problem is to consider the question of estimating the number of distinct fringe subtrees in a leaf-centric binary tree source under a generalized interpretation of "distinctness", as will be done in the following two chapters for simply generated families of trees and families of increasing trees.

Chapter 4

Simply generated families of trees

4.1 Introduction

As in the previous chapter, we investigate the number of fringe subtrees in random rooted trees in this chapter. The random tree model we consider in this chapter are *random simply generated trees* as a general concept to model uniform probability distributions, among others, on various families of trees (a formal definition follows in Section 4.2). For example, the uniform distribution on the set $\mathcal{T}_n$ of plane trees of size n and the uniform distribution on the set of d-ary trees of size n can be modeled using simply generated families of trees.

The number of distinct fringe subtrees in random simply generated trees has already been studied in [96] by Ralaivaosaona and Wagner, who showed that the expected number of distinct fringe subtrees in a random simply generated tree of size n is asymptotically equal to $c \cdot n/\sqrt{\log n}$, where the constant c depends on the particular family of trees (their result generalizes earlier results from [38] in the context of simply generated families of trees). In particular, the results in [38, 96] cover for example uniformly random plane trees (where the constant c evaluates to $c = \sqrt{2/\pi}$) and uniformly random binary trees (with $c = 2\sqrt{2/\pi}$).

In [38, 96] (as well as in [9, 12, 24, 36] and [S8, S9]), the number of distinct fringe subtrees is counted under the particular interpretation of distinctness that two trees are considered as distinct if they are distinct as members of the particular family of trees. In this chapter, we investigate the number of distinct fringe subtrees with respect to random simply generated trees *under a generalized notion of distinctness*, which allows for many different interpretations of what "distinct" trees are.

To give a concrete example of different notions of distinctness, consider the family of d-ary trees (Definition 2.5), where each node has d possible positions to which children can be attached (for instance, if $d = 3$, a left, a middle and a right position). The following three possibilities lead to different interpretations

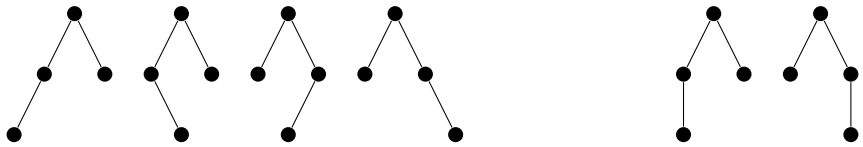


Figure 4.1: Four distinct binary trees (left), and the two distinct plane trees associated to them (right), which are in turn identical as unordered trees

of when two trees are regarded the same:

(i) the order and the positions of branches matter,

(ii) the order of branches matters, but not the positions to which they are attached,

(iii) neither the order nor the positions matter.

See Figure 4.1 and Figure 4.2 for an illustration: In Figure 4.2, we consider a binary tree (on the left) and its distinct fringe subtrees (on the right) under the three different interpretations (i) – (iii) of distinctness. In case (i) (the order and the position of branches matter), we count distinct binary fringe subtrees, in case (ii) (only the order of branches matters) we count distinct plane fringe subtrees, and in the last case (iii) (neither order nor positions matter), we count distinct unordered fringe subtrees of the binary tree.

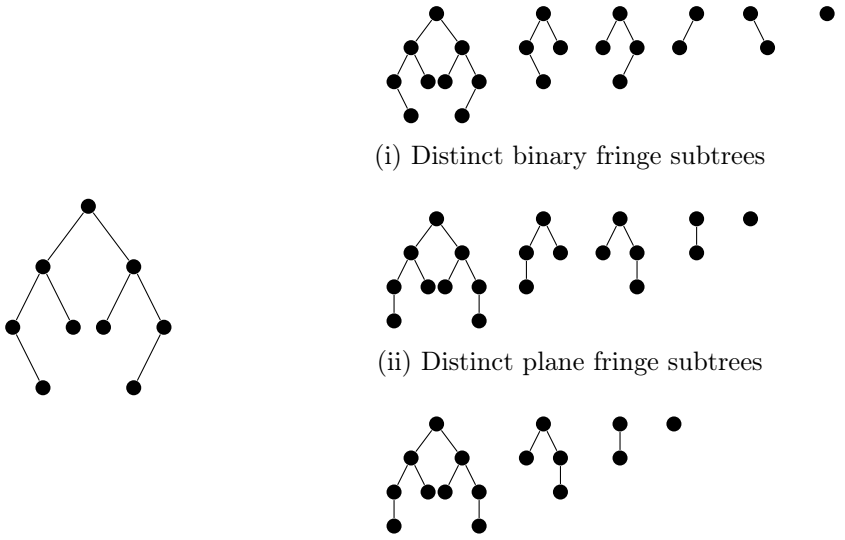
In order to cover all these cases, we only assume that the trees of size k within the given family of trees are partitioned into a set $\mathscr{I}_k$ of isomorphism classes for every k. The quantity of interest is the total number of isomorphism classes that occur among the fringe subtrees of a random tree with n nodes.

As a general main theorem of this chapter, we prove that under rather mild assumptions on the partition into isomorphism classes, the number of isomorphism classes that occur among the fringe subtrees of a random simply generated tree with n nodes is in $\Theta(n/\sqrt{\log n})$, both in expectation and w.h.p. (with high probability, i.e., with probability tending to 1 as $n \to \infty$). The precise statement is given in Theorem 4.13 in Section 4.3, the conditions we assume on the isomorphism classes are given in (C1) and (C2) in Section 4.3.

As a main application, we then count the numbers of distinct fringe subtrees in random simply generated trees under the three different notions of distinctness (i)-(iii) for several particular families of random trees in Section 4.4, Section 4.5 and Section 4.6.

In particular, we settle an open conjecture from [75], where it is stated as an open problem to estimate the expected number of distinct unordered trees represented by the fringe subtrees of a uniformly random binary tree of size n and conjectured that this number asymptotically grows as $\Theta(n/\sqrt{\log n})$. Indeed, we show that the number of distinct unordered fringe subtrees F_n in a uniformly random binary tree of size n is asymptotically bounded by

$$\underline{c} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)) \leq F_n \leq \overline{c} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1))$$



(i) Distinct binary fringe subtrees

(ii) Distinct plane fringe subtrees

(iii) Distinct unordered fringe subtrees

Figure 4.2: A binary tree (left) and (i) the six distinct binary trees, (ii) the five distinct plane trees and (iii) the four distinct unordered trees represented by its fringe subtrees (right)

for the constants $\underline{c} \approx 1.2721401445$ and $\overline{c} \approx 1.2925885353$, both in expectation and w.h.p. (Corollary 4.26).

Moreover, we refine the results from [12, 38, 96] by showing that the asymptotic estimate $c \cdot n/\sqrt{\log n}$ (where the constant c depends on the particular family of trees) on the number of distinct (as members of the concrete family) fringe subtrees in a random simply genertated tree does not only hold in expectation, but also w.h.p.

The results presented in this chapter appeared in [S11] (see also the conference version [S10]).

4.2 Simply generated families of trees and Galton–Watson trees

Let $n_0^t, \dots, n_{|t|}^t$ denote the numbers of nodes of degree i for $0 \le i \le |t|$ in a tree t. A general concept to model probability distributions on various families of trees is the concept of *simply generated families of trees*. It was introduced by Meir and Moon in [82] (see also [27, 58]). The main idea is to assign a weight to every plane tree $t \in \mathcal{T}$ which depends on the numbers $n_0^t, \dots, n_{|t|}^t$. Let $(\phi_i)_{i \ge 0}$ denote a sequence of non-negative real numbers (called the *weight sequence*), and let

$$\Phi(x) = \sum_{i \ge 0} \phi_i x^i.$$

We define the *weight* $w(t)$ of a plane tree $t \in \mathcal{T}$ as

$$w(t) = \prod_{v \in V(t)} \phi_{\deg(v)} = \prod_{i \geq 0} \phi_i^{n_i^t}.$$

Moreover, let

$$w_n = \sum_{t \in \mathcal{T}_n} w(t)$$

denote the sum of all weights of plane trees of size n. A weight sequence $(\phi_i)_{i \geq 0}$ induces a probability mass function $P_\Phi \colon \mathcal{T}_n \to [0,1]$ on the set of plane trees of size n by $P_\Phi(t) = w(t)/w_n$ for every n with $w_n > 0$. We will tacitly assume that $w_n > 0$ holds whenever we consider random trees of size n. Several families of trees can be modeled as simply generated families of trees.

Example 4.1. The family of plane trees $\mathcal{T}$ is the simply generated family of trees with weight sequence $(\phi_i)_{i \geq 0}$ defined by $\phi_i = 1$ for every $i \geq 0$. Thus, every plane tree $t \in \mathcal{T}$ is assigned the weight $w(t) = 1$, the numbers w_n count the number of distinct plane trees of size n, and the probability mass function $P_\Phi \colon \mathcal{T}_n \to [0,1]$ specifies the uniform probability distribution on $\mathcal{T}_n$.

Example 4.2. The family of d-ary trees is obtained as the simply generated family of trees whose weight sequence $(\phi_i)_{i \geq 0}$ satisfies $\phi_i = \binom{d}{i}$ for every $i \geq 0$. This takes into account that there are $\binom{d}{i}$ many types of nodes of degree i in d-ary trees. The weight $w(t)$ of a plane tree t then equals the number of distinct d-ary trees with plane representation t and the numbers w_n count the number of distinct d-ary trees of size n. For $d = 2$ we obtain the family of binary trees $\mathcal{B}^\diamond$ (Definition 2.3). The family of full binary trees $\mathcal{B}$ (Definition 2.2) corresponds to the weight sequence $(\phi_i)_{i \geq 0}$ with $\phi_0 = \phi_2 = 1$ and $\phi_i = 0$ for $i \notin \{0, 2\}$.

Example 4.3. A *Motzkin tree* is an ordered rooted tree such that each node has either zero, one or two children. In particular, we do not distinguish between left-unary and right-unary nodes as in the case of binary trees, i.e., there is only one type of unary nodes. The weight sequence $(\phi_i)_{i \geq 0}$ with $\phi_0 = \phi_1 = \phi_2 = 1$ and $\phi_i = 0$ for $i \geq 3$ corresponds to the simply generated family of Motzkin trees and the probability mass function $P_\Phi \colon \mathcal{T}_n \to [0,1]$ corresponds to the uniform probability distribution on the set of Motzkin trees of size n.

Example 4.4. Also, the family of numbered trees (Definition 2.10) can be modeled as a simply generated family of trees. Given a numbered tree (which is unorderd by definition), there are $\prod_{v \in V(t)} \deg(v)!$ possibilities to define an ordering on its nodes, that is, $\prod_{v \in V(t)} \deg(v)!$ ordered trees correspond to the same numbered tree. Furthermore, there are $n!$ possibilites to label a plane tree of size n in such a way that every node obtains a label from the set $\{1, \ldots, n\}$ and such that no two nodes obtain the same label. The family of numbered trees is obtained as the simply generated family of trees whose weight sequence $(\phi_i)_{i \geq 0}$ satisfies $\phi_i = 1/i!$ for every $i \geq 0$ (see [27, 58]). Thus, the weight of a

plane tree t equals $(\prod_{v\in V(t)} \deg(v)!)^{-1}$, and the total weight w_n of all plane trees of size n equals $\frac{1}{n!}$ times the number of numbered trees of size n.

Example 4.5. Let Σ be a finite alphabet of size $|\Sigma| = \sigma$. The family of Σ-labeled plane trees $\mathcal{T}(\Sigma)$ is obtained via the weight sequence $(\phi_i)_{i\geq 0}$ defined by $\phi_i = \sigma$ for every $i \geq 0$. In the same way, we obtain the family of Σ-labeled binary trees $\mathcal{B}^{\circ}(\Sigma)$ by setting $\phi_0 = \sigma$, $\phi_1 = 2\sigma$ and $\phi_2 = \sigma$. In particular, if all the weights of a weight sequence $(\phi_i)_{i\geq 0}$ are integers, we get a combinatorial interpretation of the corresponding simply generated family of trees as follows: trees from this family are labeled plane trees, such that nodes of degree i are assigned a label from an alphabet of size ϕ_i.

We denote a simply generated family of trees with $\mathscr{F}$. The set of trees of size k from the simply generated family $\mathscr{F}$ will be denoted by $\mathscr{F}_k$. Throughout this chapter, we denote with $R > 0$ the radius of convergence of the series $\Phi(x) = \sum_{i\geq 0} \phi_i x^i$. Furthermore, we assume that there is a real number $\tau \in (0, R]$ which satisfies $\tau\Phi'(\tau) = \Phi(\tau)$ and $\Phi''(\tau) < \infty$. We have the following theorem on the asymptotic growth of the numbers w_n:

Theorem 4.6 ([27], Theorem 3.6 and Remark 3.7)**.** *Let* $\gcd(\Phi)$ *denote the greatest common divisor of all indices* i *with* $\phi_i > 0$ *of the weight-generating series* Φ*. Then*

$$w_n = \gcd(\Phi) \cdot \sqrt{\frac{\Phi(\tau)}{2\pi\Phi''(\tau)}} \frac{\Phi'(\tau)^n}{n^{3/2}} \left(1 + \mathcal{O}(n^{-1})\right),$$

if $n \equiv 1 \mod \gcd(\Phi)$*, and* $w_n = 0$ *if* $n \not\equiv 1 \mod \gcd(\Phi)$*.*

For the sake of simplicity, we will tacitly assume that $\gcd(\Phi) = 1$ holds for the simply generated families of trees considered in this chapter, though all results presented below can be easily shown to hold for $\gcd(\Phi) \neq 1$ as well.

Galton–Watson processes. Closely related to the concept of simply generated families of trees is the concept of Galton–Watson processes.

Definition 4.7 (Galton–Watson process.)**.** Let ξ be a non-negative integer-valued random variable (called an *offspring distribution*). A *Galton–Watson branching process* (see for example [58]) with offspring distribution ξ assigns a probability $\nu(t)$ to a plane tree $t \in \mathcal{T}$ by

$$\nu(t) = \prod_{v\in V(t)} \mathbb{P}(\xi = \deg(v)) = \prod_{i\geq 0} \mathbb{P}(\xi = i)^{n_i^t}.$$

A Galton–Watson processes generates a random plane tree T as follows. In a top-down way, starting at the root node, we determine for each node v of T independently its degree $\deg(v)$ according to the distribution ξ. The probability that $\deg(v) = i$ for some integer i is given by $\mathbb{P}(\xi = i)$. If $\deg(v) = i > 0$, we attach i new nodes to v and the process continues at these newly attached nodes.

If $\deg(v) = 0$, the process stops at this node. It is thus convenient to assume that $\mathbb{P}(\xi = 0) > 0$. Note that this process might generate infinite trees with non-zero probability.

A random plane tree T_ξ generated by a Galton–Watson process is called an *unconditioned Galton–Watson tree*. Conditioning the Galton–Watson tree on the event that $|T_\xi| = n$, we obtain a probability mass function P_ξ on the set $\mathcal{T}_n$ of plane trees of size n defined by

$$P_\xi(t) = \frac{\nu(t)}{\sum_{t' \in \mathcal{T}_n} \nu(t')}.$$

A random variable $T_{n,\xi}$ which takes values in $\mathcal{T}_n$ according to the probability mass function P_ξ is called a *conditioned Galton–Watson tree* of size n. If ξ is clear from the context, we often write T_n instead of $T_{n,\xi}$ for a conditioned Galton–Watson tree of size n and T instead of T_ξ for the corresponding unconditioned Galton–Watson tree. A Galton–Watson process with offspring distribution ξ that satisfies $\mathbb{E}(\xi) = 1$ is called *critical*.

Relation between simply generated families of trees and Galton–Watson processes. Let $\mathscr{F}$ be a simply generated family of trees with weights $(\phi_i)_{i\geq 0}$. In many cases, it is possible to view a random tree of size n drawn from $\mathcal{T}_n$ according to the probability mass function P_Φ as a conditioned Galton–Watson tree (see for example [58]). Define an offspring distribution ξ by

$$\mathbb{P}(\xi = i) = \phi_i \tau^i \Phi(\tau)^{-1} \tag{4.1}$$

for every $i \geq 0$. This is well-defined, as

$$\sum_{i\geq 0} \mathbb{P}(\xi = i) = \sum_{i\geq 0} \frac{\phi_i \tau^i}{\Phi(\tau)} = \frac{\Phi(\tau)}{\Phi(\tau)} = 1, \tag{4.2}$$

and furthermore, we have

$$\mathbb{E}(\xi) = \sum_{i\geq 0} i\mathbb{P}(\xi = i) = \sum_{i\geq 0} \frac{i\phi_i \tau^i}{\Phi(\tau)} = \frac{\tau\Phi'(\tau)}{\Phi(\tau)} = 1. \tag{4.3}$$

Then ξ is an offspring distribution of a critical Galton–Watson process. In particular, ξ defined as in (4.1) induces the same probability mass function on $\mathcal{T}_n$ as the weight sequence $(\phi_i)_{i\geq 0}$, since we have $P_\xi(t) = P_\Phi(t)$. Hence, many results proved in the context of Galton–Watson trees become applicable in the setting of simply generated families of trees. Regarding the variance of the offspring distribution ξ of a Galton–Watson process corresponding to a simply generated family of trees $\mathscr{F}$ with weight sequence $(\phi_i)_{i\geq 0}$, we find

$$\mathbb{V}(\xi) = \frac{\tau^2\Phi''(\tau)}{\Phi(\tau)}. \tag{4.4}$$

Note that if $\tau < R$, then $\mathbb{V}(\xi) < \infty$, but if $\tau = R$, $\mathbb{V}(\xi)$ might be infinite. However, we will only consider weight sequences $(\phi_i)_{i\geq 0}$ for which the corresponding offspring distribution ξ satisfies $\mathbb{V}(\xi) < \infty$.

Example 4.8. For the family of plane trees, we have $\Phi(x) = \sum_{i\geq 0} x^i$. We find that $\tau = 1/2$ solves the equation $\tau\Phi'(\tau) = \Phi(\tau)$. Thus, the offspring distribution ξ of the Galton–Watson process corresponding to the family of plane trees is given by $\mathbb{P}(\xi = i) = 2^{-i-1}$ for every $i \geq 0$ (a geometric distribution).

Example 4.9. For the family of d-ary trees, we find that $\Phi(x) = (1 + x)^d$ and $\tau = (d - 1)^{-1}$. The offspring distribution of the Galton–Watson process corresponding to the family of d-ary trees is given by $\mathbb{P}(\xi = i) = \binom{d}{i} d^{-d}(d-1)^{d-i}$ for $0 \leq i \leq d$ (a binomial distribution).

Example 4.10. In the case of Motzkin trees, we have $\Phi(x) = 1 + x + x^2$ and $\tau = 1$. The Galton–Watson process with offspring distribution ξ defined by $\mathbb{P}(\xi = i) = 1/3$ if $0 \leq i \leq 2$ and $\mathbb{P}(\xi = i) = 0$ otherwise corresponds to the family of Motzkin trees.

Example 4.11. We obtain $\Phi(x) = e^x$ for the family of numbered trees. The equation $\tau\Phi'(\tau) = \Phi(\tau)$ is solved by $\tau = 1$ in this case. The Galton–Watson process corresponding to the family of numbered trees is defined by the offspring distribution ξ with $\mathbb{P}(\xi = i) = (ei!)^{-1}$ for every $i \geq 0$ (a Poisson distribution).

Additive functionals in Galton–Watson trees. Let $f\colon \mathcal{T} \to \mathbb{R}$ denote a function mapping a plane tree to a real number (called a *toll-function*). We define a mapping $F\colon \mathcal{T} \to \mathbb{R}$ by

$$F(t) = \sum_{v\in V(t)} f(t[v]).$$

Such a mapping F is called an *additive functional*. Equivalently, F can be defined by a recursion. If $t_1, t_2, \ldots, t_i$ are the root branches of t (the components resulting when the root is removed), then

$$F(t) = f(t) + \sum_{j=1}^{i} F(t_j).$$

The following theorem follows from Theorem 1.3 and Remark 5.3 in [59].

Theorem 4.12 ([59], Theorem 1.3 and Remark 5.3)**.** *Let ξ be an offspring distribution with $\mathbb{E}(\xi) = 1$, and let T_n denote the corresponding conditioned Galton–Watson tree of size n and T the corresponding unconditioned Galton–Watson tree. If $\mathbb{E}(|f(T)|) < \infty$ and $|\mathbb{E}(f(T_k))| = o(k^{1/2})$, then*

$$\lim_{n\to\infty} \mathbb{P}\left(\left|\frac{F(T_n)}{n} - \mathbb{E}(f(T))\right| > \varepsilon\right) = 0$$

for every $\varepsilon > 0$.

4.3 A general main theorem

In this section, we prove our main theorem of this chapter on the number of distinct fringe subtrees in a random simply generated tree under a generalized notion of "distinctness". As mentioned in Section 4.1, we assume that the trees of size k within the given family $\mathscr{F}$ of trees are partitioned into a set $\mathscr{I}_k$ of isomorphism classes for every k. The quantity of interest is the total number of isomorphism classes that occur among the fringe subtrees of a random tree with n nodes. The following rather mild assumptions turn out to be sufficient for our purposes:

(C1) We have $\limsup_{k\to\infty} \frac{\log|\mathscr{I}_k|}{k} = C_1 < \infty$.

(C2) There exist subsets $\mathscr{J}_k \subseteq \mathscr{I}_k$ of isomorphism classes and a positive constant C_2 such that

  (C2a) a random tree in the family $\mathscr{F}$ with k nodes belongs to a class in $\mathscr{J}_k$ with probability $1 - o(1)$ as $k \to \infty$, and

  (C2b) the probability that a random tree in $\mathscr{F}$ with k nodes lies in a fixed class $I \in \mathscr{J}_k$ is never greater than $2^{-C_2 k + o(k)}$.

Note that (C2a) and (C2b) imply that $|\mathscr{I}_k| \geq |\mathscr{J}_k| \geq 2^{C_2 k - o(k)}$, thus we have $C_1 \geq C_2 > 0$. Under the conditions (C1) and (C2), we prove the following general statement.

Theorem 4.13. *Let $\mathscr{F}$ be a simply generated family of trees with a partition into isomorphism classes that satisfies (C1) and (C2), and let ξ be the offspring distribution of the Galton–Watson process corresponding to $\mathscr{F}$, which satisfies $\mathbb{E}(\xi) = 1$ and $\mathbb{V}(\xi) < \infty$. Let F_n denote the total number of different isomorphism classes represented by the fringe subtrees of a random tree T_n of size n drawn randomly from the specific family $\mathscr{F}$. We have*

(i) $$\frac{\sqrt{2C_2}}{\sqrt{\pi\mathbb{V}(\xi)}} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)) \leq \mathbb{E}(F_n) \leq \frac{\sqrt{2C_1}}{\sqrt{\pi\mathbb{V}(\xi)}} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)),$$

(ii) $$\frac{\sqrt{2C_2}}{\sqrt{\pi\mathbb{V}(\xi)}} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)) \leq F_n \leq \frac{\sqrt{2C_1}}{\sqrt{\pi\mathbb{V}(\xi)}} \cdot \frac{n}{\sqrt{\log n}}(1 + o(1)) \text{ w.h.p.}$$

In short, in order to prove Theorem 4.13, we again make use of the *cut-point technique* that was already used in the previous chapter in order to obtain results on the number of distinct fringe subtrees in random trees (see Section 3.2), together with a refinement that takes our generalized notion of distinctness into account, and in combination with an inclusion-exclusion-like argument as in the proof of Theorem 3.33, in order to get the lower bound. Also in a similar way as in Chapter 3, we start with estimates on the number of all fringe subtrees of size equal to, respectively, larger than k in a random simply generated tree of size n. For this, we start with the following lemma.

Lemma 4.14. *Let $Z_{n,k}$ be the number of fringe subtrees of size k in a conditioned Galton–Watson tree of size n whose offspring distribution ξ satisfies $\mathbb{E}(\xi) = 1$ and $\mathbb{V}(\xi) < \infty$. Then we have*

$$\mathbb{E}(Z_{n,k}) = \frac{n}{\sqrt{2\pi\mathbb{V}(\xi)}k^{3/2}}(1 + o(1)), \tag{4.5}$$

and $\mathbb{V}(Z_{n,k}) = \mathcal{O}(n/k^{3/2})$ uniformly in k for $k \le \sqrt{n}$ as $k, n \to \infty$. Moreover, for all $k \le n$, we have

$$\mathbb{E}(Z_{n,k}) = \mathcal{O}\Big(\frac{n^{3/2}}{k^{3/2}(n-k+1)^{1/2}}\Big). \tag{4.6}$$

Proof. Let S_n be the sum of n independent copies of the offspring distribution: $S_n = \sum_{i=1}^{n} \xi_i$. By [59, Lemma 5.1], we have

$$\mathbb{E}(Z_{n,k}) = \frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)} q_k n,$$

where q_k is the probability that an *unconditioned* Galton–Watson tree with offspring distribution ξ has final size k. Moreover, by [59, Lemma 5.2], we have

$$\frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)} = 1 + \mathcal{O}\Big(\frac{k}{n}\Big) + o(n^{-1/2})$$

uniformly for all k with $1 \le k \le \frac{n}{2}$ as $n \to \infty$. Furthermore, by [59, Eq. (4.13)] (see also [69]), we find that

$$q_k \sim \frac{1}{\sqrt{2\pi\mathbb{V}(\xi)}} k^{-3/2} \tag{4.7}$$

as $k \to \infty$. Combining these results, we obtain the desired asymptotic formula (4.5) for $\mathbb{E}(Z_{n,k})$ if $k \le \sqrt{n}$ and both k and n tend to infinity. For arbitrary k, [59, Lemma 5.2] states that

$$\frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)} = \mathcal{O}\Big(\frac{n^{1/2}}{(n-k+1)^{1/2}}\Big).$$

The estimate (4.6) follows. For the variance, we can similarly employ the result [59, Lemma 6.1], which gives us

$$\begin{aligned}\mathbb{V}(Z_{n,k}) = {} & \frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)} q_k n - \Big(\frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)}\Big)^2 q_k^2 n(2k-1) \\ & + \Big(\frac{\mathbb{P}(S_{n-2k} = n-2k+1)}{\mathbb{P}(S_n = n-1)} - \Big(\frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)}\Big)^2\Big) q_k^2 n(n-2k+1).\end{aligned}$$

Finally, by [59, Lemma 6.2],

$$\frac{\mathbb{P}(S_{n-2k} = n-2k+1)}{\mathbb{P}(S_n = n-1)} - \Big(\frac{\mathbb{P}(S_{n-k} = n-k)}{\mathbb{P}(S_n = n-1)}\Big)^2 = \mathcal{O}\Big(\frac{1}{n}\Big)$$

for $k \leq \sqrt{n}$, uniformly in k. Combining all estimates, we find that the variance satisfies $\mathbb{V}(Z_{n,k}) = \mathcal{O}(q_k n) = \mathcal{O}(n/k^{3/2})$, which completes the proof. □

We remark that identity (4.5) also follows from a result in [16, Lemma 4.6] combined with the asymptotics (4.7) on the probability that an unconditioned Galton–Watson tree is of size k. From Lemma 4.14, we can now derive the following lemma on fringe subtrees of a random tree T_n of size n drawn from a simply generated family $\mathscr{F}$.

Lemma 4.15. *Let T_n be a random tree of size n drawn randomly from a simply generated family of trees $\mathscr{F}$ such that the offspring distribution ξ of the corresponding critical Galton–Watson process satisfies $\mathbb{V}(\xi) < \infty$. Let c, ε be positive real numbers with $\varepsilon < \frac{1}{2}$. For positive integers k, let $\mathscr{S}_k \subseteq \mathscr{F}_k$ be a subset of trees of size k from $\mathscr{F}$, and let p_k be the probability that a random tree of size k from the given family $\mathscr{F}$ belongs to $\mathscr{S}_k$. Let $X_{n,k}$ denote the (random) number of fringe subtrees of size k in the random tree T_n which belong to $\mathscr{S}_k$. Moreover, let $Y_{n,\varepsilon}$ denote the (random) number of arbitrary fringe subtrees of size greater than n^ε in T_n. Then*

(a) $\mathbb{E}(X_{n,k}) = p_k n (2\pi \mathbb{V}(\xi) k^3)^{-1/2}(1 + o(1))$, *for all k with $c \log n \leq k \leq n^\varepsilon$.*

(b) $\mathbb{V}(X_{n,k}) = \mathcal{O}(p_k n / k^{3/2})$ *for all k with $c \log n \leq k \leq n^\varepsilon$.*

(c) $\mathbb{E}(Y_{n,\varepsilon}) = \mathcal{O}(n^{1-\varepsilon/2})$, *and*

(d) with high probability, the following statements hold simultaneously:

(i) $|X_{n,k} - \mathbb{E}(X_{n,k})| \leq p_k^{1/2} n^{1/2+\varepsilon} k^{-3/4}$ *for all k with $c \log n \leq k \leq n^\varepsilon$,*

(ii) $Y_{n,\varepsilon} \leq n^{1-\varepsilon/3}$.

We emphasize (since it will be important later) that the inequality in part (d), item (i), does not only hold w.h.p. for each individual k, but that it is satisfied w.h.p. for all k in the given range simultaneously. Parts (a) and (b) were shown in the context of Galton–Watson trees in [16, Lemma 4.6 and Lemma 4.8].

Proof. Let $Z_{n,k}$ as in Lemma 4.14 denote the number of fringe subtrees of size k in the conditioned Galton–Watson tree of size n with offspring distribution ξ. Then by the correspondence between simply generated families of trees and conditioned Galton–Watson trees, we find that $Z_{n,k}$ and the random number of fringe subtrees of size k in a random tree T_n of size n drawn randomly from the simply generated family $\mathscr{F}$ are identically distributed. Furthermore, conditioned on $Z_{n,k} = N$ for some integer N, the N fringe subtrees of size k in T_n are independent random trees in $\mathscr{F}_k$ with the same distribution. Thus, $X_{n,k}$ can be regarded as a sum of $Z_{n,k}$ many Bernoulli random variables with probability p_k. We thus have (see [51, Theorem 15.1, p.84])

$$\mathbb{E}(X_{n,k}) = p_k \mathbb{E}(Z_{n,k}) = \frac{n p_k}{\sqrt{2\pi \mathbb{V}(\xi)} k^{3/2}}(1 + o(1)),$$

as well as (see again [51, Theorem 15.1, p.84])

$$\mathbb{V}(X_{n,k}) = p_k^2 \mathbb{V}(Z_{n,k}) + p_k(1-p_k)\mathbb{E}(Z_{n,k}) = \mathcal{O}\left(\frac{np_k}{k^{3/2}}\right)$$

by Lemma 4.14, which proves part (a) and part (b). For part (c), we observe that

$$\mathbb{E}(Y_{n,\varepsilon}) = \sum_{k>n^\varepsilon} \mathbb{E}(Z_{n,k}) = \mathcal{O}\left(n^{1-\varepsilon/2}\right),$$

again by Lemma 4.14. In order to show part (d), we apply Chebyshev's inequality to obtain concentration on $X_{n,k}$:

$$\mathbb{P}\left(|X_{n,k} - \mathbb{E}(X_{n,k})| \geq p_k^{1/2} n^{1/2+\varepsilon} k^{-3/4}\right) \leq \frac{\mathbb{V}(X_{n,k})}{p_k n^{1+2\varepsilon} k^{-3/2}} = \mathcal{O}(n^{-2\varepsilon}).$$

Hence, by the union bound, the probability that the stated inequality fails for any k in the given range is only $\mathcal{O}(n^{-\varepsilon})$, proving that the first statement holds w.h.p. Finally, Markov's inequality implies that

$$\mathbb{P}\left(Y_{n,\varepsilon} > n^{1-\varepsilon/3}\right) \leq \frac{\mathbb{E}(Y_{n,\varepsilon})}{n^{1-\varepsilon/3}} = \mathcal{O}(n^{-\varepsilon/6}),$$

showing that the second inequality holds w.h.p. as well. □

We are now able to prove Theorem 4.13. The proof consists of two parts. First, the upper bound is verified; then we prove the lower bound, which has the same order of magnitude. As mentioned above, we again make use of the *cut-point technique*, which was already used for similar purposes in [24, 36, 38, 96] and in the proofs of Theorem 3.10, Theorem 3.15 and Theorem 3.20 in Chapter 3. For the lower bound, we use an argument based on the inclusion-exclusion-principle. A basic variant of this proof technique already appeared in [96] and in the proof of Theorem 3.33.

Proof of Theorem 4.13. The upper bound. For some integer k_0 (to be specified later), we can clearly bound the total number of isomorphism classes covered by the fringe subtrees of a random tree T_n of size n from above by the sum of

(i) the total number of isomorphism classes of trees of size smaller than k_0, which is $\sum_{k<k_0} |\mathscr{I}_k|$ (a deterministic quantity that does not depend on the tree T_n), and

(ii) the total number of fringe subtrees of T_n of size greater than or equal to k_0.

To estimate the number (i) of isomorphism classes of trees of size smaller than k_0, we note that $|\mathscr{I}_k| \leq 2^{C_1 k + o(k)}$ by condition (C1), thus also

$$\sum_{k<k_0} |\mathscr{I}_k| \leq 2^{C_1 k_0 + o(k_0)}. \tag{4.8}$$

Therefore, we can choose $k_0 = k_0(n)$ in such a way that $k_0 = \frac{\log n}{C_1} - g(n)$ for a function g with $g(n) = o(\log n)$ and

$$\sum_{k<k_0} |\mathscr{I}_k| = o\Big(\frac{n}{\sqrt{\log n}}\Big), \tag{4.9}$$

thus making this part negligible. The concrete choice of the function g depends on the lower-order term in the exponent on the right-hand side of (4.8), and furthermore, g has to be chosen large enough, so that the bound $o(n/\sqrt{\log n})$ on the sum of sizes of isomorphism classes in (4.9) is achieved. For our purposes, it is enough to note that there exists such a function g.

In order to estimate the number (ii) of fringe subtrees of T_n of size greater than or equal to k_0, we apply Lemma 4.15 with $\varepsilon = 1/6$. We let $\mathscr{S}_k$ be the set of all trees of size k generated by our simply generated family of trees, so that $p_k = 1$, to obtain the upper bound

$$\begin{aligned}\sum_{k_0\le k\le n^\varepsilon} X_{n,k} + Y_{n,\varepsilon} &= \frac{n}{\sqrt{2\pi\mathbb{V}(\xi)}} \sum_{k_0\le k\le n^\varepsilon} \frac{1}{k^{3/2}}\,(1+o(1)) + \mathcal{O}\left(n^{1-\varepsilon/3}\right)\\ &= \frac{2}{\sqrt{2\pi\mathbb{V}(\xi)}}\frac{n}{\sqrt{k_0}} + o\left(\frac{n}{\sqrt{\log n}}\right),\end{aligned}$$

in expectation and w.h.p. as well, as the estimate from Lemma 4.15 (part (d)) holds w.h.p. simultaneously for all k in the given range. Now we combine the two bounds to obtain the upper bound on F_n stated in Theorem 4.13, both in expectation and w.h.p.

The lower bound. Let $\mathscr{S}_k$ now be the set of trees that belong to isomorphism classes in $\mathscr{J}_k$ (see condition (C2)). Our lower bound is based on counting only fringe subtrees which belong to $\mathscr{S}_k$ for suitable k. By condition (C2a), we know that the probability p_k that a random tree in $\mathscr{F}$ conditioned on having size k belongs to a class in $\mathscr{J}_k$ tends to 1 as $k \to \infty$. Hence, by Lemma 4.15, we find that the number of fringe subtrees of size k in T_n that belong to $\mathscr{S}_k$ is

$$X_{n,k} = \frac{n}{\sqrt{2\pi\mathbb{V}(\xi)k^3}}(1+o(1)),$$

both in expectation and w.h.p. We show that most of these trees are the only representatives of their isomorphism classes as fringe subtrees. We choose a cut-point $k_1 = k_1(n)$; the precise choice will be described later. For $k \ge k_1$, let $X^{(2)}_{n,k}$ denote the (random) number of unordered pairs of isomorphic trees (trees belonging to the same isomorphism class) among the fringe subtrees of size k which belong to $\mathscr{S}_k$. We will determine an upper bound for its expected value. To this end, let i_k denote the number of isomorphism classes of trees in $\mathscr{S}_k$, and let $r_1, r_2, \dots, r_{i_k}$ be the probabilities that a random tree of size k lies in the respective classes. By condition (C2b), we have $r_i \le 2^{-C_2 k + o(k)}$ for every $1 \le i \le i_k$. Let us condition on the event that $X_{n,k} = N$ for some integer $0 \le N \le n$. Those N fringe subtrees are all independent random trees. Thus,

for each of the $\binom{N}{2}$ pairs of fringe subtrees, the probability that both belong to the i-th isomorphism class is r_i^2. This gives us

$$\mathbb{E}(X_{n,k}^{(2)} \mid X_{n,k} = N) = \binom{N}{2} \sum_{i=1}^{i_k} r_i^2 \leq \frac{n^2}{2} \sum_{i=1}^{i_k} r_i 2^{-C_2 k + o(k)} \leq \frac{n^2}{2} 2^{-C_2 k + o(k)}.$$

Since this holds for all N, the law of total expectation yields

$$\mathbb{E}(X_{n,k}^{(2)}) \leq \frac{n^2}{2} 2^{-C_2 k + o(k)}.$$

Summing over $k \geq k_1$, we find that

$$\sum_{k \geq k_1} \mathbb{E}(X_{n,k}^{(2)}) \leq \frac{n^2}{2} \sum_{k \geq k_1} 2^{-C_2 k + o(k)} \leq \frac{n^2}{2} 2^{-C_2 k_1 + o(k_1)}. \tag{4.10}$$

We can therefore choose k_1 in such a way that $k_1 = \frac{\log n}{C_2} + g(n)$, again for a function g with $g(n) = o(\log n)$ and such that

$$\sum_{k \geq k_1} \mathbb{E}(X_{n,k}^{(2)}) = o\Big(\frac{n}{\sqrt{\log n}}\Big). \tag{4.11}$$

Here again the concrete choice of the function g depends on the lower-order term in the exponent on the right-hand side of (4.10), and furthermore, g has to be chosen large enough, so that the bound $o(n/\sqrt{\log n})$ on the sum of sizes of isomorphism classes in (4.11) is achieved. If an isomorphism class of trees of size k occurs i times among the fringe subtrees of a random tree of size n, it contributes $i - \binom{i}{2}$ to the random variable $X_{n,k} - X_{n,k}^{(2)}$. As $i - \binom{i}{2} \leq 1$ for all non-negative integers i, we find that $X_{n,k} - X_{n,k}^{(2)}$ is a lower bound on the total number of isomorphism classes covered by fringe subtrees of T_n. This gives us

$$F_n \geq \sum_{k_1 \leq k \leq n^\varepsilon} X_{n,k} - \sum_{k_1 \leq k \leq n^\varepsilon} X_{n,k}^{(2)},$$

where we choose ε as in the proof of the upper bound. The second sum is negligible since it is $o(n/\sqrt{\log n})$ in expectation and thus also w.h.p. by the Markov inequality. For the first sum, the same calculation as for the upper bound (using Lemma 4.15) shows that it is

$$\frac{2n}{\sqrt{2\pi \mathbb{V}(\xi) k_1}} + o\left(\frac{n}{\sqrt{\log n}}\right)$$

both in expectation and w.h.p. This proves Theorem 4.13. □

4.4 Distinct fringe subtrees

In this section, we count distinct fringe subtrees in random trees T_n of size n drawn from a simply generated family of trees $\mathscr{F}$, where we consider two subtrees as distinct, if they are distinct as members of the particular family $\mathscr{F}$ of trees. For this, we apply Theorem 4.13 with the following partition into isomorphism classes $\mathscr{I}$: we consider two trees as isomorphic if they are identical as members of $\mathscr{F}$, that is, each tree is isomorphic only to itself. The total number of isomorphism classes $|\mathscr{I}_k|$ is thus the total number of trees in $\mathscr{F}$ of size k. In order to ensure that condition (C1) from Theorem 4.13 is satisfied, we need to make an additional assumption on $\mathscr{F}$. We assume that the weights ϕ_i of the weight sequence $(\phi_i)_{i\geq 0}$ are integers, and that each tree $t \in \mathscr{F}$ corresponds to a weight of one unit, such that the total weight w_n of all plane trees of size n then equals the number of distinct trees of size n in the simply generated family $\mathscr{F}$ of trees. This assumption is satisfied, e.g., by the simply generated family of plane trees (Example 4.1 and Example 4.8), the family of d-ary trees (Example 4.2 and Example 4.9), the family of Motzkin trees (Example 4.3 and Example 4.10) and the families of Σ-labeled plane trees and Σ-labeled binary trees (Example 4.5).

We obtain the following result from Theorem 4.13.

Theorem 4.16. *Let F_n denote the total number of distinct fringe subtrees in a random tree T_n of size n from a simply generated family $\mathscr{F}$ of trees with weight-generating series Φ whose weights ϕ_i are integers. Then we have*

(i) $$\mathbb{E}(F_n) = \frac{2}{\tau}\sqrt{\frac{\Phi(\tau)\log(\Phi'(\tau))}{2\pi\Phi''(\tau)}} \cdot \frac{n}{\sqrt{\log n}}(1+o(1)),$$

(ii) $$F_n = \frac{2}{\tau}\sqrt{\frac{\Phi(\tau)\log(\Phi'(\tau))}{2\pi\Phi''(\tau)}} \cdot \frac{n}{\sqrt{\log n}}(1+o(1)) \text{ w.h.p.}$$

The first part (i) of Theorem 4.16 was already shown in [38, 96], our new contribution is part (ii).

Proof. We verify that the conditions of Theorem 4.13 are satisfied if we consider the partition of $\mathscr{F}$ into isomorphism classes of size one. We find that

$$|\mathscr{I}_k| = w_k,$$

i.e., the number $|\mathscr{I}_k|$ of isomorphism classes of trees of size k equals the number w_k of distinct trees of size k in the respective simply generated family of trees $\mathscr{F}$. With Theorem 4.6, we have

$$|\mathscr{I}_k| = \sqrt{\frac{\Phi(\tau)}{2\pi\Phi''(\tau)}}\frac{\Phi'(\tau)^k}{k^{3/2}}(1+\mathcal{O}(k^{-1})),$$

so condition (C1) is satisfied with $C_1 = \log(\Phi'(\tau))$. In order to show that condition (C2) holds, define $\mathscr{J}_k = \mathscr{I}_k$, so that every random tree of size k in the family $\mathscr{F}$ belongs to a class in $\mathscr{J}_k$, and the probability that a random tree in $\mathscr{F}$

of size k lies in a fixed isomorphism class $I \in \mathscr{J}_k$ is $1/w_k$. Thus, condition (C2) holds as well, and we have $C_2 = C_1 = \log(\Phi'(\tau))$. Recall that by (4.4), we find that the variance of the Galton–Watson process corresponding to $\mathscr{F}$ is given by $\mathbb{V}(\xi) = \tau^2\Phi''(\tau)/\Phi(\tau)$. Theorem 4.16 now follows from Theorem 4.13. □

For some concrete simply generated families of trees, we obtain the following results from Theorem 4.16.

Corollary 4.17. *Let F_n denote the total number of distinct fringe subtrees in a uniformly random plane tree of size n. Then*

(i) $\mathbb{E}(F_n) = \sqrt{\frac{2}{\pi}}\frac{n}{\sqrt{\log n}}(1 + o(1))$,

(ii) $F_n = \sqrt{\frac{2}{\pi}}\frac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

Proof. Recall that the family of plane trees is obtained as the simply generated family of trees with weight sequence $(\phi_i)_{i\geq 0}$ with $\phi_i = 1$ for every $i \geq 0$ (see Examples 4.1 and 4.8). In particular, we find that $\Phi(x) = \sum_{k\geq 0} x^k$ and that $\tau = \frac{1}{2}$ solves the equation $\tau\Phi'(\tau) = \Phi(\tau)$. Thus, the leading constant in Theorem 4.16 evaluates to

$$\frac{2}{\tau}\sqrt{\frac{\Phi(\tau)\log(\Phi'(\tau))}{2\pi\Phi''(\tau)}} = \sqrt{\frac{2}{\pi}}.$$

This proves the statement. □

Corollary 4.18. *Let F_n denote the total number of distinct fringe subtrees in a uniformly random d-ary tree of size n. Then for the constant*

$$c_d = \left(\frac{2d}{\pi}\Big(\frac{d}{d-1}\log d - \log(d-1)\Big)\right)^{1/2}$$

we find that

(i) $\mathbb{E}(F_n) = c_d \cdot \frac{n}{\sqrt{\log n}}(1 + o(1))$,

(ii) $F_n = c_d \cdot \frac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

In particular, we get the constant $c_2 = 2\sqrt{2/\pi}$ in the case of binary trees.

Proof. For the family of d-ary trees (Examples 4.2 and 4.9) we find that the function Φ is given by $\Phi(x) = (1 + x)^d$ and that $\tau = (d - 1)^{-1}$ satisfies the equation $\tau\Phi'(\tau) = \Phi(\tau)$. Therefore, the leading constant in Theorem 4.16 evaluates to

$$\frac{2}{\tau}\sqrt{\frac{\Phi(\tau)\log(\Phi'(\tau))}{2\pi\Phi''(\tau)}} = \left(\frac{2d}{\pi}\Big(\frac{d}{d-1}\log d - \log(d-1)\Big)\right)^{1/2}.$$

This proves the statement. □

In the same way, we obtain from Theorem 4.16 results on the number of distinct fringe subtrees in uniformly random Σ-labeled plane trees and uniformly random Σ-labeled binary trees (see Example 4.5). The leading constants evaluate in these cases to $\sqrt{\log(4\sigma)/\pi}$, respectively, $2\sqrt{\log(4\sigma)/\pi}$, where $\sigma = |\Sigma|$. Thus, Theorem 4.16 in particular also generalizes results shown in [12].

We remark that Theorem 4.16 does not apply to the family of numbered trees (see Example 4.4), as the weight sequence corresponding to the family of numbered trees is not a sequence of integers. In particular, the number of numbered trees of size n is n^{n-1} (see for example [27]), and thus, a partition of the set of numbered trees of size n into isomorphism classes of size one does not satisfy condition (C1) from Theorem 4.13.

The total number F_n of distinct fringe subtrees in a uniformly random numbered tree of size n was estimated in [96], where it was shown that

$$\mathbb{E}(F_n) = \sqrt{\frac{2}{\pi}}\frac{n\sqrt{\log(\ln n)}}{\sqrt{\log n}}\left(1 + \mathcal{O}\left(\frac{\log\log\log n}{\log\log n}\right)\right).$$

Here, two fringe subtrees are considered the same if there is an isomorphism that preserves the relative order of the labels. Note that numbered trees are called "labeled trees" in [96].

4.5 Plane fringe subtrees

In this section, we consider simply generated families $\mathscr{F}$ of trees which admit a plane embedding. For instance, for the family of d-ary trees (see Example 4.2), we find that each d-ary tree can be considered as a plane tree in a natural way by simply forgetting the positions to which the branches of the nodes are attached, such that there is no distinction between different types of nodes of the same degree. Likewise, trees from the simply generated family of numbered trees (see Example 4.4) admit a unique plane representation if we order the children of each node according to their labels and then disregard the node labels. If a family $\mathscr{F}$ admits a plane embedding, it is possible to count the number of fringe subtrees which are distinct as plane trees. For the family of plane trees (see Example 4.1), the results from this section will be equivalent to the results presented in the previous section. We start with the following lemma.

Lemma 4.19. *Let ξ be the offspring distribution of a critical Galton–Watson process satisfying $\mathbb{V}(\xi) < \infty$, and let T_k be a conditioned Galton–Watson tree of size k with respect to ξ. Let $M = \{i \in \mathbb{N}_0 \mid \mathbb{P}(\xi = i) > 0\}$, and let*

$$\mu = \sum_{i \in M} \mathbb{P}(\xi = i)\log(\mathbb{P}(\xi = i)).$$

The probability that $\nu(T_k) \leq 2^{(\mu+\varepsilon)k}$ tends to 1 for every fixed $\varepsilon > 0$ as $k \to \infty$.

Proof. For the proof, we make use of Theorem 4.12 on additive functionals in Galton–Watson trees (see Section 4.2). Let $\deg_r(t)$ denote the degree of the root

node of a plane tree $t \in \mathcal{T}$, and define the toll function $f\colon \mathcal{T} \to \mathbb{R}$ by

$$f(t) = \begin{cases} \log(\mathbb{P}(\xi = \deg_r(t))) & \text{if } \mathbb{P}(\xi = \deg_r(t)) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

For every $t \in \mathcal{T}$ with $\nu(t) > 0$, the associated additive functional is

$$\begin{aligned} F(t) &= \sum_{v \in V(t)} f(t[v]) = \sum_{v \in V(t)} \log\left(\mathbb{P}(\xi = \deg_r(t[v]))\right) \\ &= \log\Big(\prod_{v \in V(t)} \mathbb{P}(\xi = \deg(v)) \Big) = \log(\nu(t)). \end{aligned}$$

If T denotes the unconditioned Galton–Watson tree corresponding to ξ, then

$$\mathbb{E}(|f(T)|) = \sum_{i \in M} \mathbb{P}(\xi = i) |\log(\mathbb{P}(\xi = i))|,$$

as the probability that the root node of an unconditioned Galton–Watson tree T has degree i for some $i \in M$ is given by $\mathbb{P}(\xi = i)$. Note that if $\mathbb{P}(\xi = i) > 2^{-i}$, we have $|\log(\mathbb{P}(\xi = i))| \le i$. Furthermore, if $\mathbb{P}(\xi = i) \le 2^{-i}$, we find that $\mathbb{P}(\xi = i)|\log(\mathbb{P}(\xi = i))| \le 2^{-i/2+1}$. Thus, we are able to bound $\mathbb{E}(|f(T)|)$ from above by

$$\mathbb{E}(|f(T)|) \le \sum_{i \ge 0} \mathbb{P}(\xi = i) i + \sum_{i \ge 0} 2^{-i/2+1} = \mathbb{E}(\xi) + 2 + 2\sqrt{2} < \infty, \tag{4.12}$$

as the Galton–Watson process is critical by assumption. Furthermore, we have

$$|\mathbb{E}(f(T_k))| \le \sum_{i \ge 0} \mathbb{P}(\deg_r(T_k) = i) |\log(\mathbb{P}(\xi = i))|.$$

By (2.7) in [57], there is a constant $c > 0$ (independent of k and i) such that

$$\mathbb{P}(\deg_r(T_k) = i) \le ci\mathbb{P}(\xi = i)$$

for all $i, k \ge 0$. We thus find

$$\begin{aligned} |\mathbb{E}(f(T_k))| &\le c \sum_{i \in M} i\mathbb{P}(\xi = i) |\log(\mathbb{P}(\xi = i))| \\ &\le c \sum_{i \ge 0} \mathbb{P}(\xi = i) i^2 + c \sum_{i \ge 0} i 2^{-i/2+1} < \infty, \end{aligned} \tag{4.13}$$

as $\mathbb{V}(\xi) < \infty$ by assumption. As the upper bound holds independently of k, we thus have $|\mathbb{E}(f(T_k))| = \mathcal{O}(1)$. Altogether, we find that the requirements of Theorem 4.12 are satisfied. Let

$$\mu = \mathbb{E}(f(T)) = \sum_{i \in M} \mathbb{P}(\xi = i) \log(\mathbb{P}(\xi = i)).$$

Then by Theorem 4.12, the probability that

$$F(T_n) = \log(\nu(T_n)) \leq (\mu + \varepsilon)n$$

holds tends to 1 for every $\varepsilon > 0$ as $n \to \infty$. Thus, the statement follows. □

Theorem 4.20. *Let F_n denote the number of distinct plane trees represented by the fringe subtrees of a random tree T_n of size n drawn from a simply generated family of trees $\mathscr{F}$ with weight-generating series Φ and let ξ denote the offspring distribution of the corresponding Galton–Watson tree. Set*

$$\kappa_\Phi = 2\tau^{-1}(\Phi(\tau))^{1/2}(2\pi\Phi''(\tau))^{-1/2}.$$

Furthermore, let $M = \{i \geq 0 \mid \phi_i > 0\}$ and define the sequence $(\widetilde{\phi}_i)_{i\geq 0}$ by $\widetilde{\phi}_i = 1$ if $i \in M$ and $\widetilde{\phi}_i = 0$ otherwise. Let $\widetilde{\Phi}(x) = \sum_{i\geq 0} \widetilde{\phi}_i x^i$, and let $\widetilde{\tau}$ denote the solution to the equation $\widetilde{\tau}\widetilde{\Phi}'(\widetilde{\tau}) = \widetilde{\Phi}(\widetilde{\tau})$. Set

$$C_1 = \log(\widetilde{\Phi}'(\widetilde{\tau})) \quad \text{and} \quad C_2 = -\mu,$$

with μ defined as in Lemma 4.19. Then

(i) $\kappa_\Phi \sqrt{C_2}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq \mathbb{E}(F_n) \leq \kappa_\Phi \sqrt{C_1}\dfrac{n}{\sqrt{\log n}}(1 + o(1)),$

(ii) $\kappa_\Phi \sqrt{C_2}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq F_n \leq \kappa_\Phi \sqrt{C_1}\dfrac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

Proof. Here we thus consider two trees as isomorphic if their plane representations are identical. This yields a partition of $\mathscr{F}_k$ into isomorphism classes $\mathscr{I}_k$, for which we verify that the conditions of Theorem 4.13 are satisfied. The number $|\mathscr{I}_k|$ of isomorphism classes equals the number of all plane trees of size k with node degrees in M, which can be determined from Theorem 4.6. The weight sequence $(\widetilde{\phi}_i)_{i\geq 0}$ characterizes the simply generated family of plane trees with node degrees in M. We thus find by Theorem 4.6:

$$\log(|\mathscr{I}_k|) = \log(\widetilde{\Phi}'(\widetilde{\tau}))k(1 + o(1)),$$

so condition (C1) of Theorem 4.13 is satisfied with

$$C_1 = \log(\widetilde{\Phi}'(\widetilde{\tau})).$$

Next, we show that condition (C2) is satisfied as well. By Lemma 4.19 (and the correspondence between simply generated trees and Galton–Watson trees), there exists a sequence of integers k_j such that

$$\mathbb{P}\big(\nu(T_k) \leq 2^{(\mu+1/j)k}\big) \geq 1 - \frac{1}{j}$$

for all $k \geq k_j$. So if we set $\varepsilon_k = \min\{\frac{1}{j} \mid k_j \leq k\}$, then

$$\mathbb{P}\big(\nu(T_k) \leq 2^{(\mu+\varepsilon_k)k}\big) \geq 1 - \varepsilon_k,$$

and $\varepsilon_k \to 0$ as $k \to \infty$. We define the subset $\mathscr{J}_k \subseteq \mathscr{I}_k$ as the set of isomorphism classes of trees whose corresponding plane embedding t satisfies $\nu(t) \le 2^{(\mu+\varepsilon_k)k}$. The probability that a random tree of size k in $\mathscr{F}$ lies in an isomorphism class in the set $\mathscr{J}_k$ is precisely the probability that a conditioned Galton–Watson tree T_k corresponding to the offspring distribution ξ satisfies $\nu(T_k) \le 2^{(\mu+\varepsilon_k)k}$ and hence tends to 1 as $k \to \infty$. Furthermore, the probability that a random tree T_k of size k in $\mathscr{F}$ has the shape of $t \in \mathcal{T}_k$ when regarded as a plane tree, i.e., the probability that T_k lies in the fixed isomorphism class $I \in \mathscr{J}_k$ containing all trees in the family $\mathscr{F}$ with plane representation t is never greater than

$$P_\xi(t) = \frac{\nu(t)}{\sum_{t' \in \mathcal{T}_k} \nu(t')}.$$

In particular, the numerator is bounded by $2^{(\mu+\varepsilon_k)k}$ as $I \in \mathscr{J}_k$. In order to estimate the denominator, we apply Theorem 4.6. We find that $\sum_{t' \in \mathcal{T}_n} \nu(t')$ is the total weight of all plane trees of size n with respect to the weight sequence $(\mathbb{P}(\xi = k))_{k \ge 0} = (\phi_k \tau^k \Phi(\tau)^{-1})_{k \ge 0}$. We thus obtain from Theorem 4.6 that

$$\sum_{t' \in \mathcal{T}_n} \nu(t') = \sqrt{\frac{\Phi(\tau)}{2\pi\tau^2\Phi''(\tau)}}\, n^{-3/2}(1 + O(1/n)). \tag{4.14}$$

Hence,

$$P_\xi(t) \le \sqrt{\frac{2\pi\tau^2\Phi''(\tau)}{\Phi(\tau)}}\, k^{3/2} 2^{(\mu+\varepsilon_k)k}(1 + O(k^{-1})) = 2^{\mu k + o(k)},$$

which shows that condition (C2) is satisfied with $C_2 = -\mu$. The statement of Theorem 4.20 now follows from Theorem 4.13 and (4.4). □

We remark that for the family of plane trees, the statement of Theorem 4.20 is equivalent to the statement of Theorem 4.16. As $\phi_i = 1$ for every $i \ge 0$ in this case, the constant C_1 in the upper bound of Theorem 4.20 evaluates to $\log(\Phi'(\tau))$. Furthermore, for every plane tree t of size n, we have $\nu(t)/\sum_{t' \in \mathcal{T}_n} \nu(t') = 1/w_n$, so that the constant C_2 in Theorem 4.20 evaluates to $\log(\Phi'(\tau))$ as well.

Let us determine the constants appearing in the upper and lower bound explicitly in some examples.

Corollary 4.21. *Let F_n denote the number of distinct plane trees represented by the fringe subtrees of a uniformly random binary tree of size n. Let*

$$\underline{c} = \sqrt{\frac{6}{\pi}} \approx 1.3819766 \text{ and } \overline{c} = \frac{2\sqrt{\log 3}}{\sqrt{\pi}} \approx 1.4205763.$$

Then

(i) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \le \mathbb{E}(F_n) \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1))$,

(ii) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \le F_n \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

Proof. The family of binary trees $\mathcal{B}^\diamond$ is obtained from the weight generating function $\Phi(x) = 1 + 2x + x^2$. A plane representation of a binary tree from $\mathcal{B}^\diamond$ is a Motzkin tree (see Example 4.3), so we find that $\widetilde{\Phi}(x) = 1 + x + x^2$ (with $\widetilde{\Phi}$ defined as in Theorem 4.20). Thus, $\widetilde{\tau} = 1$ solves the equation $\widetilde{\tau}\widetilde{\Phi}'(\widetilde{\tau}) = \widetilde{\Phi}(\widetilde{\tau})$ and $\widetilde{\Phi}'(\widetilde{\tau}) = 3$. Hence, the constant C_1 in Theorem 4.20 evaluates to $C_1 = \log 3$. We remark that the asymptotic growth of the number of Motzkin trees is well known, see e.g. [37]. To compute the constant for the lower bound, we find that $\tau = 1$ and $\Phi(\tau) = \Phi'(\tau) = 4$. Hence, the offspring distribution ξ of the Galton–Watson process corresponding to $\mathcal{B}^\diamond$ is defined by $\mathbb{P}(\xi = 0) = 1/4$, $\mathbb{P}(\xi = 1) = 1/2$ and $\mathbb{P}(\xi = 2) = 1/4$. We find

$$\mu = \sum_{k=0}^{2} \mathbb{P}(\xi = k) \log(\mathbb{P}(\xi = k)) = -\frac{3}{2},$$

and hence $C_2 = 3/2$. With $\kappa_\Phi = 2\tau^{-1}(\Phi(\tau))^{1/2}(2\pi\Phi''(\tau))^{-1/2} = 2/\sqrt{\pi}$, the statement follows. □

Similarly, for the family of numbered trees, we get the following result (recall that we obtain a unique plane representation of a numbered tree if we order the children of each node according to their labels and then disregard the labels).

Corollary 4.22. *Let F_n denote the number of distinct plane trees represented by the fringe subtrees of a uniformly random numbered tree of size n. Let*

$$\underline{c} = \left(\frac{2}{\pi e} \sum_{k\geq 2} \frac{\log(e) + \log(k!)}{k!}\right)^{1/2} \approx 1.0947286$$

and $\overline{c} = \sqrt{\frac{4}{\pi}} \approx 1.1283792$. Then

(i) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq \mathbb{E}(F_n) \leq \overline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1))$,

(ii) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq F_n \leq \overline{c}\dfrac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

Proof. The family of numbered trees is obtained as the simply generated family of trees with weight sequence $(\phi_k)_{k\geq 0}$ satisfying $\phi_k = 1/k!$ for every $k \geq 0$ (see Examples 4.4 and 4.11). We find that $\widetilde{\Phi}(x) = \sum_{k\geq 0} x^k$ and that $\widetilde{\tau} = 1/2$ solves the equation $\widetilde{\tau}\widetilde{\Phi}'(\widetilde{\tau}) = \widetilde{\Phi}(\widetilde{\tau})$, so that the constant C_1 in Theorem 4.13 evaluates to $C_1 = \log 4 = 2$. In order to compute the constant for the lower bound, we first notice that $\tau = 1$ solves the equation $\tau\Phi'(\tau) = \Phi(\tau)$ with $\Phi(\tau) = e$. The offspring distribution ξ of the Galton–Watson process corresponding to the family of numbered trees is well known to be the Poisson distribution (with $\mathbb{P}(\xi = k) = (ek!)^{-1}$ for every $k \geq 0$). Hence, we have

$$\mu = \sum_{k\geq 0} \mathbb{P}(\xi = k) \log(\mathbb{P}(\xi = k)) = -e^{-1} \sum_{k\geq 0} \frac{\log(e) + \log(k!)}{k!}.$$

With $\kappa_\Phi = 2\tau^{-1}(\Phi(\tau))^{1/2}(2\pi\Phi''(\tau))^{-1/2} = \sqrt{2/\pi}$, the statement follows. □

4.6 Unordered fringe subtrees

In this section, we apply Theorem 4.13 in order to count the number of distinct unordered trees represented by the fringe subtrees of a random tree of size n drawn randomly from a simply generated family of trees. Thus we consider two trees from the family $\mathscr{F}$ as isomorphic if their unordered representations are identical. This is meaningful for all simply generated families, since every rooted tree has a natural unordered representation. Let $t \in \mathcal{T}$ be a plane tree. As a simple application of the orbit-stabilizer theorem [5, Proposition 6.9.2], one finds that the number of plane trees with the same unordered representation as t is given by $\prod_{v \in V(t)} \deg(v)! / |\operatorname{Aut}(t)|$, where $|\operatorname{Aut}(t)|$ denotes the cardinality of the automorphism group of t. This is because the permutations of the branches at the different nodes of t generate a group of order $\prod_{v \in V(t)} \deg(v)!$ acting on the plane trees with the same unordered representation as t, and $|\operatorname{Aut} t|$ is the subgroup that fixes t. It follows that

$$\nu(t) \frac{\prod_{v \in V(t)} \deg(v)!}{|\operatorname{Aut}(t)|}$$

is the total weight of all plane representations of t within a simply generated family. This quantity will play the same role that $\nu(t)$ played in the proof of Theorem 4.20. From Theorem 4.12, we obtain the following result.

Lemma 4.23. *Let ξ be the offspring distribution of a critical Galton–Watson process satisfying $\mathbb{V}(\xi) < \infty$, and let T_k be a conditioned Galton–Watson tree of size k with respect to ξ. Then there is a constant $\widehat{\mu} < 0$ such that the probability that*

$$\nu(T_k) \frac{\prod_{v \in V(T_k)} \deg(v)!}{|\operatorname{Aut}(T_k)|} \leq 2^{(\widehat{\mu}+\varepsilon)k}$$

holds tends to 1 for every $\varepsilon > 0$ as $k \to \infty$.

Proof. As in the proof of Lemma 4.19, we aim to define a suitable additive functional and apply Theorem 4.12. To this end, we need a recursive description of $|\operatorname{Aut} t|$, the size of the automorphism group. Let $\deg_r(t)$ again denote the degree of the root node of t, let $t_1, t_2, \ldots, t_{\deg_r(t)}$ be the root branches of a tree t, and let $m_1, m_2, \ldots, m_{k_t}$ denote the multiplicities of isomorphic branches of t ($m_1 + m_2 + \cdots + m_{k_t} = \deg_r(t)$). Here we call two trees isomorphic if they are identical as unordered trees. That is, the $\deg_r(t)$ many subtrees rooted at the children of the root node fall into k_t many different isomorphism classes, where m_i of them belong to isomorphism class i, respectively. Then we have

$$|\operatorname{Aut}(t)| = \prod_{j=1}^{\deg_r(t)} |\operatorname{Aut}(t_j)| \cdot \prod_{i=1}^{k_t} m_i!\,,$$

since an automorphism of t acts as an automorphism within branches and also possibly permutes branches that are isomorphic. In fact, the whole structure of

$\operatorname{Aut}(t)$ is well understood [61]. It follows from the recursion for $|\operatorname{Aut}(t)|$ that

$$F(t) = \log\left(\frac{\nu(t)\prod_{v\in V(t)}\deg(v)!}{|\operatorname{Aut}(t)|}\right)$$

(well-defined for all t with $\nu(t) > 0$) is the additive functional associated with the toll function f that is defined by

$$f(t) = \log(\mathbb{P}(\xi = \deg_r(t))\deg_r(t)!) - \log(m_1!\cdots m_{k_t}!), \tag{4.15}$$

if $\mathbb{P}(\xi = \deg_r(t)) > 0$, and $f(t) = 0$ otherwise. Let $M = \{i \geq 0 \mid \mathbb{P}(\xi = i) > 0\}$, and let T be the unconditioned Galton–Watson tree corresponding to ξ. Since $0 \leq \log(\deg_r(t)!) - \log(m_1!m_2!\cdots m_{k_t}!) \leq \log(\deg_r(t)!)$, we have

$$\mathbb{E}(|f(T)|) \leq \sum_{i\in M}\mathbb{P}(\xi = i)|\log(\mathbb{P}(\xi = i))| + \sum_{i\in M}\mathbb{P}(\xi = i)|\log(i!)|.$$

The first sum was shown to be finite earlier in (4.12), and the second sum is finite as $\log(i!) = \mathcal{O}(i^2)$ and $\mathbb{V}(\xi) < \infty$ by assumption. Moreover, we find

$$|\mathbb{E}(f(T_k))| \leq \sum_{\substack{i\in M\\ i\leq k}}\mathbb{P}(\deg_r(T_k) = i)|\log(\mathbb{P}(\xi = i)i!)|.$$

By (2.7) in [57], there is a constant $c > 0$ (independent of k and i) such that

$$\mathbb{P}(\deg_r(T_k) = i) \leq ci\mathbb{P}(\xi = i)$$

for all $i, k \geq 0$. We thus find

$$\begin{aligned}|\mathbb{E}(f(T_k))| &\leq c\sum_{\substack{i\in M\\ i\leq k}} i\mathbb{P}(\xi = i)|\log\left(\mathbb{P}(\xi = i)i!\right)|\\ &\leq c\sum_{i\in M} i\mathbb{P}(\xi = i)|\log\left(\mathbb{P}(\xi = i)\right)| + c\sum_{\substack{i\in M\\ i\leq k}} i\mathbb{P}(\xi = i)\log\left(i!\right).\end{aligned}$$

The first sum was shown to be finite in (4.13). As $\log(i!) \leq i\log i$, we obtain

$$\sum_{\substack{i\in M\\ i\leq k}} i\mathbb{P}(\xi = i)\log\left(i!\right) \leq \log k\sum_{i\in M} i^2\mathbb{P}(\xi = i) = \mathcal{O}(\log k),$$

for the second sum, as by assumption, $\mathbb{V}(\xi) < \infty$. In particular, we thus have $\mathbb{E}|f(T_k)| = \mathcal{O}(\log k)$. Altogether, we find that the requirements of Theorem 4.12 are satisfied. Now set $\widehat{\mu} = \mathbb{E}(f(T))$. By Theorem 4.12, the probability that

$$F(T_k) = \log\left(\frac{\nu(T_k)\prod_{v\in V(T_k)}\deg(v)!}{|\operatorname{Aut}(T_k)|}\right) \leq (\widehat{\mu} + \varepsilon)k$$

holds tends to 1 for every $\varepsilon > 0$ as $k \to \infty$. Thus, the statement follows. □

Additionally, we need the following result on the number of unordered trees with node degrees from some given set $M \subseteq \mathbb{N}_0$:

Theorem 4.24 ([37, pp. 71-72])**.** *Let $M \subseteq \mathbb{N}_0$ with $0 \in M$, and let u_k^M denote the number of unordered rooted trees t of size k with the property that the degree of every node in t lies in M. Then*

$$u_k^M \sim a_M \cdot \frac{b_M^k}{k^{3/2}}$$

if $k \equiv 1 \mod \gcd(M)$, where $\gcd(M)$ is the greatest common divisor of all elements of M, and $u_k^M = 0$ otherwise, where a_M, b_M depend on M.

Again for the sake of simplicity, we assume that $\gcd(M) = 1$ holds for all families of trees considered in the following. We are now able to derive the following theorem.

Theorem 4.25. *Let F_n denote the total number of distinct unordered trees represented by the fringe subtrees of a random tree T_n of size n drawn from a simply generated family of trees $\mathscr{F}$. Set $\kappa_\Phi = 2\tau^{-1}(\Phi(\tau))^{1/2}(2\pi\Phi''(\tau))^{-1/2}$. Furthermore, let $M = \{i \in \mathbb{N}_0 \mid \phi_i > 0\}$ and set $C_1 = \log(b_M)$, where b_M is the constant in Theorem 4.24, and $C_2 = -\widehat{\mu}$, where $\widehat{\mu}$ is the constant in Lemma 4.23. Then*

(i) $\kappa_\Phi \sqrt{C_2} \dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq \mathbb{E}(F_n) \leq \kappa_\Phi \sqrt{C_1} \dfrac{n}{\sqrt{\log n}}(1 + o(1))$,

(ii) $\kappa_\Phi \sqrt{C_2} \dfrac{n}{\sqrt{\log n}}(1 + o(1)) \leq F_n \leq \kappa_\Phi \sqrt{C_1} \dfrac{n}{\sqrt{\log n}}(1 + o(1))$ *w.h.p.*

Proof. Here we consider two trees as isomorphic if their unordered representations are identical. This yields a partition of $\mathscr{F}_k$ into isomorphism classes $\mathscr{I}_k$, for which we verify that the conditions of Theorem 4.13 are satisfied. The number $|\mathscr{I}_k|$ of isomorphism classes equals the number of all unordered trees of size k with node degrees in M, which is given by Theorem 4.24. We have

$$\log(|\mathscr{I}_k|) = \log(b_M)k(1 + o(1)).$$

Hence, condition (C1) is satisfied with $C_1 = \log(b_M)$. Note that if two plane trees $t, t' \in \mathcal{T}$ have the same unordered representation, we have $\nu(t) = \nu(t')$, $\prod_{v \in V(t)} \deg(v)! = \prod_{v \in V(t')} \deg(v)!$ and $|\operatorname{Aut}(t)| = |\operatorname{Aut}(t')|$. As in the proof of Theorem 4.20, we can now use Lemma 4.23 to show that there exists a sequence ε_k that tends to 0 as $k \to \infty$ with the property that

$$\mathbb{P}\Big(\nu(T_k) \frac{\prod_{v \in V(T_k)} \deg(v)!}{|\operatorname{Aut}(T_k)|} \leq 2^{(\widehat{\mu} + \varepsilon_k)k}\Big) \geq 1 - \varepsilon_k.$$

So let $\mathscr{J}_k \subseteq \mathscr{I}_k$ denote the subset of isomorphism classes of trees in $\mathscr{F}_k$ such that the trees t that they represent satisfy

$$\nu(t) \frac{\prod_{v \in V(t)} \deg(v)!}{|\operatorname{Aut}(t)|} \leq 2^{(\widehat{\mu} + \varepsilon_k)k}.$$

The probability that a random tree of size k drawn from $\mathscr{F}_k$ lies in an isomorphism class that belongs to the set $\mathscr{J}_k$ is precisely the probability that a conditioned Galton–Watson tree T_k of size k with offspring distribution ξ satisfies

$$\nu(T_k)\frac{\prod_{v\in V(T_k)}\deg(v)!}{|\operatorname{Aut}(T_k)|} \le 2^{(\widehat{\mu}+\varepsilon_k)k},$$

which is at least $1-\varepsilon_k$ by construction. Thus condition (C2a) is satisfied. In order to show that condition (C2b) is satisfied as well, let $I \in \mathscr{J}_k$ be a single isomorphism class, and let u be the unordered tree that it represents. The probability that a random tree in $\mathscr{F}$ of size k lies in the isomorphism class I is

$$\frac{\nu(u)}{\sum_{t\in\mathcal{T}_k}\nu(t)}\frac{\prod_{v\in u}\deg(v)!}{|\operatorname{Aut}(u)|},$$

since $\prod_{v\in t}\deg(v)!/|\operatorname{Aut}(t)|$ equals the number of plane representations of the tree u, each of which has probability $\nu(u)$. As in the proof of Theorem 4.20 (see (4.14)), we have

$$\sum_{t\in\mathcal{T}_k}\nu(t) = \sqrt{\frac{\Phi(t)}{2\pi\tau^2\Phi''(\tau)}}k^{-3/2}(1+O(k^{-1})).$$

Thus, the probability that a random tree in $\mathscr{F}$ of size k lies in a single isomorphism class $I \in \mathscr{J}_k$ is never greater than

$$\sqrt{\frac{2\pi\tau^2\Phi''(\tau)}{\Phi(\tau)}}k^{3/2}2^{(\widehat{\mu}+\varepsilon_k)k}(1+O(k^{-1})) = 2^{\widehat{\mu}k+o(k)}.$$

So condition (C2b) is satisfied as well, with $C_2 = -\widehat{\mu}$. Theorem 4.25 now follows directly from Theorem 4.13. □

In order to obtain bounds on the number F_n of distinct unordered trees represented by the fringe subtrees of a random tree T_n drawn from some concrete family of trees, we need to determine the values of the constants $\widehat{\mu}$ and b_M in Lemma 4.23 and Theorem 4.24 for the particular family of trees. We show this in two examples.

Corollary 4.26. *Let F_n denote the number of distinct unordered trees represented by the fringe subtrees of a uniformly random binary tree with n nodes. Then for $\underline{c} \approx 1.2721401445$ and $\overline{c} \approx 1.2925885353$, we have*

(i) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1)) \le \mathbb{E}(F_n) \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1))$,

(ii) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1)) \le F_n \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1))$ *w.h.p.*

Proof. We prove the statement for the family of *full binary trees* $\mathcal{B}$. The corollary then follows from the one-to-one correspondence between the set $\mathcal{B}_n^\diamond$ of binary trees of size n and the set $\mathcal{B}_n$ of full binary trees of leafsize n, and the fact that

the number of distinct fringe subtrees in a binary tree equals the number of distinct fringe subtrees in the corresponding full binary tree minus one. For the family of full binary trees $\mathcal{B}$, the required values can be obtained from known results. The number of unordered rooted trees of leafsize k with node degrees in $M = \{0, 2\}$ is given by the k^{th} *Wedderburn-Etherington number* W_k. The asymptotic growth of these numbers is

$$W_k \sim a_M \cdot k^{-3/2} \cdot b_M^k,$$

for certain positive constants a_M, b_M [10, 35], where $b_M \approx 2.4832535363$. In order to determine a concrete value for the constant $\widehat{\mu}$ in Lemma 4.23 for the family of full binary trees $\mathcal{B}$, we make use of a theorem by Bóna and Flajolet [10] on the number of automorphisms of a uniformly random *full* binary tree. The following statement is stated for phylogenetic trees in [10], but the two probabilistic models are equivalent: Consider a uniformly random full binary tree T_k with k leaves, and let $|\operatorname{Aut}(T_k)|$ be the cardinality of its automorphism group. The logarithm of this random variable satisfies a central limit theorem: For certain positive constants α and β, we have

$$\mathbb{P}(|\operatorname{Aut}(T_k)| \leq 2^{\alpha k + \beta\sqrt{k}x}) \overset{k\to\infty}{\to} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt \tag{4.16}$$

for every real number x, where $\alpha \approx 0.2710416936$ [10].

The simply generated family $\mathcal{B}$ of full binary trees corresponds to the weight sequence with $\phi_0 = \phi_2 = 1$ and $\phi_j = 0$ for $j \notin \{0, 2\}$. The corresponding offspring distribution ξ satisfies $\mathbb{P}(\xi = 0) = \mathbb{P}(\xi = 2) = 1/2$. Let t denote a full binary tree of size $n = 2k - 1$, with k leaves and $k - 1$ internal nodes. Then $\nu(t) = 2^{-2k+1}$ and $\prod_{v \in V(t)} \deg(v)! = 2^{k-1}$, and consequently

$$\nu(T_k) \frac{\prod_{v \in V(T_k)} \deg(v)!}{|\operatorname{Aut}(T_k)|} = \frac{1}{2^k |\operatorname{Aut}(T_k)|}$$

for a random full binary tree T_k with k leaves. It follows from (4.16) that for every $\varepsilon > 0$,

$$\lim_{n\to\infty} \mathbb{P}\Big(\Big| \frac{1}{2k-1} \log\Big(\nu(T_k) \frac{\prod_{v \in V(T_k)} \deg(v)!}{|\operatorname{Aut}(T_k)|} \Big) + \frac{1+\alpha}{2} \Big| > \varepsilon \Big) = 0$$

thus $\widehat{\mu} = -\frac{1+\alpha}{2} \approx -0.6355208468$ in this special case. The statement now follows from Theorem 4.25. □

For the family of numbered trees (Example 4.4), we similarly obtain the following result from Theorem 4.25.

Corollary 4.27. *Let F_n denote the number of distinct unordered trees represented by the fringe subtrees of a uniformly random numbered tree of size n. Then for $\underline{c} \approx 0.9830940673$ and $\overline{c} \approx 0.9976849212$, we have*

(i) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1)) \le \mathbb{E}(F_n) \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1))$,

(ii) $\underline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1)) \le F_n \le \overline{c}\dfrac{n}{\sqrt{\log n}}(1+o(1))$ *w.h.p.*

Proof. For the family of numbered trees, we have $M = \{0, 1, 2, \ldots\}$, and the number of isomorphism classes is the number of *Pólya trees* (rooted unordered trees), which follows the same kind of asymptotic formula as the Wedderburn-Etherington numbers above, with a growth constant $b_M \approx 2.9557652857$, see [91], [37, Section VII.5] or [35, Section 5.6]. This immediately gives us the value of $C_1 = \log(b_M)$. The number of automorphisms satisfies a similar central limit theorem as in (4.16), with a constant $\alpha \approx 0.0754386818$ (and k being the number of nodes rather than the number of leaves), see [90]. Since we have $\mathbb{P}(\xi = i) = \frac{1}{ei!}$ for numbered trees, the expression $\log(\mathbb{P}(\xi = \deg_r(t))\deg_r(t)!)$ in (4.15) simplifies to $\log(1/e)$ for every value of $\deg_r(t)$. So we have $\widehat{\mu} = \log(1/e) - \alpha$ and thus $C_2 = -\log(1/e) + \alpha \approx 1.0754386818$. Finally, $\kappa_\Phi = \sqrt{2/\pi}$ in this example. Putting everything together, we obtain the following numerical values for the constants in Theorem 4.25 in the case of numbered trees:

$$\kappa_\Phi\sqrt{C_2} \approx 0.9830940673 \quad \text{and} \quad \kappa_\Phi\sqrt{C_1} \approx 0.9976849212.$$

The statement follows. □

4.7 Conclusion and open problems

Our main theorem of this chapter is quite general and covers many different types of trees as well as different notions of distinctness. As the examples with explicit constants show, the upper and lower bounds they provide are typically quite close.

Nevertheless, the following natural question arises from our results: In Theorem 4.20 and Theorem 4.25 respectively, we have derived upper and lower bounds on the number F_n of distinct plane, respectively, distinct unordered fringe subtrees in a random simply-generated tree T_n. Is it possible, instead of lower and upper bounds, to determine a constant c (depending on the concrete notion of distinctness) such that F_n asymptotically grows as $c \cdot n/\log n$ in expectation and with high probability? In order to prove such estimates, it seems essential to gain a better understanding of the different additive functionals that we employed in the proofs of these theorems, in particular their distributions further away from the mean values.

Moreover, we remark that to determine the variance $\mathbb{V}(F_n)$ is an open problem, even under the usual, non-generalized notion of distinctness and even for very basic probability distributions as the uniform distribution on the family of plane trees of size n.

Chapter 5

Increasing trees

5.1 Introduction

In this chapter, we again investigate the number of distinct fringe subtrees in random trees. The random tree model considered in this chapter is the model of *very simple families of increasing trees* (a formal definition follows), which in particular incorporates the binary search tree model (Definition 2.15).

In the same way as in Chapter 4, we estimate the number of distinct fringe subtrees *under a generalized notion of distinctness*. We again assume that the trees of size k within the given family of trees are partitioned into a set $\mathscr{I}_k$ of isomorphism classes for every k. The quantity of interest is then again the total number of isomorphism classes that occur among the fringe subtrees of a random tree with n nodes. This in particular incorporates the three particular notions of distinctness (i)-(iii) presented in Section 4.1.

So far, distinct fringe subtrees in random increasing trees have only been investigated under the particular notion of distinctness that two trees are considered as distinct if their shapes are distinct as members of the particular family of trees. In [9], it is shown for two particular families of increasing trees (recursive trees and binary search trees) that the average number of distinct fringe subtrees in a random tree of size n is asymptotically bounded from above by $\mathcal{O}(n/\log n)$ and bounded from below by $\Omega(\sqrt{n})$. Furthermore, it is conjectured that the asymptotics $\Theta(n/\log n)$ holds.

For the binary search tree model, it was already shown in [36, 24], that the expected number of distinct fringe subtrees in a random binary search tree of size n is upper-bounded by $4n/\log n(1+o(1))$. Moreover, Devroye showed in [24] that an asymptotic lower bound of the form $c \cdot n/\log n$ holds for the constant $c = \log(3)/2$, and in [S8] an asymptotic lower bound of the form $c \cdot n/\log n$ was shown for the constant $c \approx 0.8681$ (see also Example 3.29). That is, for the binary search tree model it is already known that the asymptotics $\Theta(n/\log n)$ holds.

Our main contribution of this chapter is a very general theorem which estimates the number of distinct fringe subtrees in a random increasing tree

(from one of the very simple families of increasing trees, that is, recursive trees, generalized plane-oriented recursive trees and d-ary increasing trees, see Section 5.2) under a generalized notion of distinctness. This general main theorem states that under rather mild assumptions on the partition into isomorphism classes, the number of isomorphism classes that occur among the fringe subtrees of such a random increasing tree with n nodes is in $\Theta(n/\log n)$, both in expectation and with high probability. The precise statement is presented in Theorem 5.3, and the conditions we assume on the isomorphism classes are given in (C1) and (C2).

The random tree model, the results and proof techniques presented in this chapter resemble in many ways the concepts, results and techniques from Chapter 4. As an application of our general main theorem, we again count the numbers of distinct fringe subtrees in random increasing trees under three particular notions of distinctness for the concrete families of increasing trees in Section 5.4, Section 5.5 and Section 5.6.

In particular, we settle the open conjecture from [9], by showing that the number of distinct fringe subtrees in a random recursive tree of size n is indeed in $\Theta(n/\log n)$, not only in expectation, but also with high probability. Furthermore, with respect to the binary search tree model, we strengthen the results from [36, 24, 9], by showing that the number of distinct (as binary trees) fringe subtrees F_n in a random binary search tree of size n satisfies

$$\frac{cn}{\log n}(1+o(1)) \leq F_n \leq \frac{4n}{\log n}(1+o(1)),$$

where $c \approx 3.472754274$, both in expectation and with high probability (see Corollary 5.6). That is, we show that the upper bound from [36, 24], does not only hold in expectation, but also with high probability, and improve the lower bound from [24], [S8].

The results presented in this chapter are published in [S11] (see also the conference version [S10]).

5.2 Very simple families of increasing trees

The random model of increasing trees is defined in a quite similar way as the model of simply generated families of trees from the previous chapter.

We call a rooted tree t *increasing*, if its nodes are labeled with the numbers $1, 2, \ldots, |t|$ in such a way that no two nodes receive the same label and that the labels along any path from the root to a leaf are increasing. A *plane-oriented recursive tree* is a labeled plane tree which is increasing. We denote the set of plane-oriented recursive trees with $\mathcal{I}$ and the set of plane-oriented recursive trees of size n with $\mathcal{I}_n$ for every $n \geq 1$.

If one assigns a weight function to plane-oriented recursive trees in the same way as to plane trees in the case of simply generated families of trees in the previous chapter, one obtains a *simple variety of increasing trees* [7, 27]. Let $n_0^t, \ldots, n_{|t|}^t$ again denote the numbers of nodes of a tree t of degree i for

$0 \leq i \leq |t|$, let $(\phi_i)_{i\geq 0}$ be a weight sequence of non-negative real numbers and define the *weight* $w(t)$ of a tree t as

$$w(t) = \prod_{v \in V(t)} \phi_{\deg(v)} = \prod_{i \geq 0} \phi_i^{n_i^t}.$$

As in the previous chapter, we obtain a probability mass function P_Φ on $\mathcal{I}_n$ for every $n \geq 1$ if we set

$$w_n = \sum_{t \in \mathcal{I}_n} w(t)$$

and $P_\Phi(t) = w(t)/w_n$. Note that P_Φ also induces a probability distribution on $\mathcal{T}_n$, if we discard the node labels of the increasing trees and identify trees of the same shape. Furthermore, we define

$$\Phi(x) = \sum_{i \geq 0} \phi_i x^i.$$

A general treatment of simple varieties of increasing trees was given by Bergeron, Flajolet and Salvy in [7]. In this chapter, we focus on three particular random models of increasing trees, which are collectively sometimes called *very simple families of increasing trees* [92]. The three random models considered in this chapter are the following.

(i) *Random plane-oriented recursive trees (ports) and random generalized plane-oriented recursive trees (gports).* A natural choice for the weight sequence $(\phi_i)_{i\geq 0}$ is to set $\phi_i = 1$ for every $i \geq 0$. This random tree model yields random plane-oriented recursive trees (ports). In this model, every tree $t \in \mathcal{I}_n$ gets a weight of one unit. The function Φ is given by $\Phi(x) = (1-x)^{-1}$ in this case. Generalized plane-oriented recursive trees (gports) are defined by $\Phi(x) = (1-x)^{-r}$ for some constant $r > 0$ (respectively, $\phi_i = \binom{r+i-1}{i}$ for every $i \geq 0$).

(ii) *Random d-ary increasing trees.* Random d-ary increasing trees are defined by $\Phi(x) = (1+x)^d$, that is, $\phi_i = \binom{d}{i}$ for every $i \geq 0$.

(iii) *Random recursive trees.* A recursive tree is an *unordered* increasing tree. The uniform distribution on the set of recursive trees of size n for every $n \geq 1$ is obtained by the weight sequence $(\phi_i)_{i\geq 0}$ with $\phi_i = 1/i!$ for every $i \geq 0$. This takes into account that for a node of degree i there are $i!$ possibilities to order its i branches. The generating series Φ is given as $\Phi(x) = e^x$ in this case.

These three models (i) - (iii) are the increasing tree analogues of (generalized) plane trees, d-ary trees and numbered trees from the previous chapter on simply generated families of trees (see Example 4.1, Example 4.2 and Example 4.4). Note that we obtain a probability distribution on the set of plane trees/d-ary trees/unordered trees of size n if we generate a random gport/d-ary increasing

tree/recursive tree of size n and then omit the node labels and identify trees of the same shape. Thus, counting distinct fringe subtrees in a random gport/d-ary increasing tree/recursive tree (under a notion of distinctness that does not consider node labels) is the same as counting distinct fringe subtrees in a random plane/d-ary/unordered tree (under the same notion of distinctness) with respect to the probability distribution induced on these sets by the respective random model of increasing trees. Let t be a plane tree with n nodes. The number of increasing labelings of the nodes with labels $1, 2, \ldots, n$ is given by

$$\frac{n!}{\prod_{v \in V(t)} |t[v]|}, \tag{5.1}$$

see for example [100, Eq. (5)] or [67, Section 5.1.4, Exercise 20]. In particular, as for two trees of size n the number of increasing labelings can differ, the probability distribution obtained on the set of plane trees/d-ary trees/unordered trees of size n by omitting the node labels of a random gport/d-ary increasing tree/recursive tree of size n does not coincide with the uniform distribution on the set of plane/d-ary/unordered trees of size n. In particular, it is well known that in the case of binary increasing trees, we obtain the binary search tree model in that way [27], that is, the random tree model of binary increasing trees and random binary search trees (Definition 2.15) are equivalent.

These three models (i), (ii) and (iii) (defined on the previous page) of increasing trees furthermore have the property that random elements from these families can be generated by a simple growth process (see e.g. [27]): We start with the root, which is labeled 1. The n-th node (labeled n) is attached at random to one of the previous $n-1$ nodes, with a probability that is proportional to a linear function of the degree of the nodes. Specifically, the probability to attach to a node v with degree i is always proportional to $1 + \alpha i$, where α is defined as follows:

- For random recursive trees, we have $\alpha = 0$. Hence, it is equally likely to attach to any of the $n-1$ existing nodes. This takes into account that recursive trees are unordered, thus, there is only one possible way to attach a new node to an existing node. Each recursive tree of size n is generated with probability of $1/(n-1)!$ in this process.

- For ports, we have $\alpha = 1$. Hence, the probability to attach to a node increases with the number of children the node has. This takes into account that ports are ordered trees, thus, if a node v already has i children, then there are $i+1$ possible ways to attach a new node to v.

  In the case of gports, we have $\alpha = 1/r$. As nodes in gports have a higher probability to become parent of a new node if they already have many children, they are also called *preferential attachment trees*.

- For d-ary increasing trees, we set $\alpha = -1/d$. Thus, nodes can only have up to d children in d-ary increasing trees, as the probability to attach further nodes becomes 0.

The number (respectively, total weight) of trees with n nodes is (see e.g. [27])

- $w_n = (n-1)!$ for recursive trees,
- $w_n = \prod_{k=1}^{n-1}(k(r+1)-1)$ for gports,
- $w_n = \prod_{k=1}^{n-1}(1+k(d-1))$ for d-ary increasing trees (in particular, $n!$ for binary increasing trees).

In order to investigate the number of distinct fringe subtrees in random trees, we again make use of results on the total number of (all) fringe subtrees of a given size in a random tree from one of the very simple families of increasing trees (the idea is then again to apply the cut-point technique as explained in Section 3.2 and as also applied in Chapter 4). The following formulas for the mean and variance are shown in [40]:

Lemma 5.1 ([40])**.** *Consider a very simple family of increasing trees. For every $k < n$, let $Z_{n,k}$ be the random number of fringe subtrees of size k in a random tree of size n drawn from the very simple family of increasing trees. Then the expectation of $Z_{n,k}$ satisfies*

$$\mathbb{E}(Z_{n,k}) = \frac{(1+\alpha)n-\alpha}{((1+\alpha)k+1)((1+\alpha)k-\alpha)},$$

and for the variance of $Z_{n,k}$, we have $\mathbb{V}(Z_{n,k}) = \mathcal{O}(n/k^2)$ uniformly in n and k.

Additive functionals in increasing trees. Furthermore, as in the previous chapter, we make use of additive functionals. Let $f\colon \mathcal{I} \to \mathbb{R}$ again denote a toll function (for our purposes, it suffices to consider toll functions whose values do not depend on the node labels). An additive tree functional is again a mapping $F\colon \mathcal{I} \to \mathbb{R}$ defined by

$$F(t) = \sum_{v \in V(t)} f(t[v]).$$

For additive functionals of increasing trees with finite support, i.e., for functionals for which there exists a constant K such that $f(t) = 0$ whenever $|t| > K$, a central limit theorem was proven in [53] and [97] (the latter even contains a slightly more general result). Those results do not directly apply to the additive functionals that we are considering here. However, convergence in probability is sufficient for our purposes. We have the following lemma:

Lemma 5.2. *Let T_n denote a random tree with n nodes from one of the very simple families of increasing trees (recursive trees, d-ary increasing trees, gports), and let F be any additive functional with toll function f. As before, set $\alpha = 0$ for recursive trees, $\alpha = -\frac{1}{d}$ for d-ary increasing trees and $\alpha = \frac{1}{r}$ for gports. We have*

$$\mathbb{E}(F(T_n)) = \mathbb{E}(f(T_n)) + \sum_{k=1}^{n-1} \frac{((1+\alpha)n-\alpha)\mathbb{E}(f(T_k))}{((1+\alpha)k+1)((1+\alpha)k-\alpha)}.$$

Moreover, if $\mathbb{E}|f(T_n)| = o(n)$ *and* $\sum_{k=1}^{\infty} \frac{\mathbb{E}|f(T_k)|}{k^2} < \infty$*, then we have*

$$\frac{F(T_n)}{n} \xrightarrow{p} \mu = \sum_{k=1}^{\infty} \frac{(1+\alpha)\mathbb{E}(f(T_k))}{((1+\alpha)k+1)((1+\alpha)k-\alpha)}.$$

Proof. The first statement follows directly from Lemma 5.1, since fringe subtrees are, conditioned on their size, again random trees following the same probabilistic model as the whole tree. For functionals with finite support, where $f(T) = 0$ for all but finitely many trees T, convergence in probability follows from the central limit theorems in [53] and [97]. For the more general case, we approximate the additive functional F with a truncated version F^m based on the toll function

$$f^m(T) = \begin{cases} f(T) & |T| \le m, \\ 0 & \text{otherwise.} \end{cases}$$

Since we already know that convergence in probability holds for functionals with finite support, we have

$$\frac{F^m(T_n)}{n} \xrightarrow{p} \mu_m = \sum_{k=1}^{m} \frac{(1+\alpha)\mathbb{E}(f(T_k))}{((1+\alpha)k+1)((1+\alpha)k-\alpha)}.$$

Now we use the triangle inequality and Markov's inequality in order to estimate $\mathbb{P}(|F(T_n)/n - \mu| > \varepsilon)$. Choose m sufficiently large so that $|\mu_m - \mu| < \frac{\varepsilon}{3}$. Then we have

$$\begin{aligned}
&\mathbb{P}\Big(\Big|\frac{F(T_n)}{n} - \mu\Big| > \varepsilon\Big) \\
&\le \mathbb{P}\Big(\Big|\frac{F^m(T_n)}{n} - \mu_m\Big| > \frac{\varepsilon}{3}\Big) + \mathbb{P}\Big(\Big|\frac{F^m(T_n) - F(T_n)}{n}\Big| > \frac{\varepsilon}{3}\Big) \\
&\le \mathbb{P}\Big(\Big|\frac{F^m(T_n)}{n} - \mu_m\Big| > \frac{\varepsilon}{3}\Big) + \frac{3}{\varepsilon}\mathbb{E}\left(\Big|\frac{F^m(T_n) - F(T_n)}{n}\Big|\right) \\
&\le \mathbb{P}\Big(\Big|\frac{F^m(T_n)}{n} - \mu_m\Big| > \frac{\varepsilon}{3}\Big) + \frac{3\mathbb{E}|f(T_n)|}{\varepsilon n} + \sum_{k=m+1}^{n-1} \frac{3((1+\alpha)n-\alpha)\mathbb{E}|f(T_k)|}{\varepsilon n((1+\alpha)k+1)((1+\alpha)k-\alpha)}
\end{aligned}$$

for $n > m$. Since $\frac{F_m(T_n)}{n} \xrightarrow{p} \mu_m$, it follows that

$$\limsup_{n\to\infty} \mathbb{P}\Big(\Big|\frac{F(T_n)}{n} - \mu\Big| > \varepsilon\Big) \le \frac{3}{\varepsilon} \sum_{k=m+1}^{\infty} \frac{(1+\alpha)\mathbb{E}|f(T_k)|}{((1+\alpha)k+1)((1+\alpha)k-\alpha)}.$$

As $\sum_{k=1}^{\infty} \frac{\mathbb{E}|f(T_k)|}{k^2} < \infty$ by assumption, we can take $m \to \infty$, and finally find that

$$\lim_{n\to\infty} \mathbb{P}\Big(\Big|\frac{F(T_n)}{n} - \mu\Big| > \varepsilon\Big) = 0,$$

completing the proof.

□

5.3 A general main theorem

As in the case of simply generated families of trees from Chapter 4, we start with a general main theorem that counts the number of distinct fringe subtrees in a random increasing tree from one of the very simple families of increasing trees under a generalized notion of distinctness.

As increasing trees of size n are labeled with the numbers $1, \dots, n$ in such a way that no two nodes obtain the same label, no two fringe subtrees of an increasing tree are identical *as labeled trees*, or in other words, the number of distinct *labeled* fringe subtrees of an increasing tree of size n is always equal to n. This leads in a natural way again to the consideration of isomorphism classes. For the sake of simplicity, we assume that the partitioning into isomorphism classes only depends on the shape of the trees and does not take node labels into account.

We again assume that the trees of size k within the given family $\mathscr{F}$ of trees are partitioned into a set $\mathscr{I}_k$ of isomorphism classes for every k. The quantity of interest is then the total number of isomorphism classes that occur among the fringe subtrees of a random tree with n nodes. We assume that the partition into isomorphism classes satisfies the same properties as in Chapter 4.3.

(C1) We have $\limsup_{k\to\infty} \frac{\log|\mathscr{I}_k|}{k} = C_1 < \infty$.

(C2) There exist subsets $\mathscr{J}_k \subseteq \mathscr{I}_k$ of isomorphism classes and a positive constant C_2 such that

- (C2a) a random tree in the family $\mathscr{F}$ with k nodes belongs to a class in $\mathscr{J}_k$ with probability $1 - o(1)$ as $k \to \infty$, and
- (C2b) the probability that a random tree in $\mathscr{F}$ with k nodes lies in a fixed isomorphism class $I \in \mathscr{J}_k$ is never greater than $2^{-C_2 k + o(k)}$.

Note that (C2a) and (C2b) imply that $|\mathscr{I}_k| \geq |\mathscr{J}_k| \geq 2^{C_2 k - o(k)}$, thus we have $C_1 \geq C_2 > 0$. Two increasing trees that have the same shape will lie in the same isomorphism class for the partitions we consider below. Under the conditions (C1) and (C2), we prove the following general statement.

Theorem 5.3. *Let $\mathscr{F}$ be one of the "very simple families" of increasing trees, namely recursive trees, d-ary increasing trees, or gports. Let a partition into isomorphism classes be given that satisfies (C1) and (C2), and let F_n denote the total number of different isomorphism classes represented by the fringe subtrees of a random tree T_n of size n drawn from $\mathscr{F}$. With $\alpha = 0$ in the case of recursive trees, $\alpha = 1/r$ in the case of gports, and $\alpha = -1/d$ for d-ary increasing trees, we have*

(i) $$\frac{C_2 n}{(1+\alpha)\log n}(1+o(1)) \leq \mathbb{E}(F_n) \leq \frac{C_1 n}{(1+\alpha)\log n}(1+o(1)),$$

(ii) $$\frac{C_2 n}{(1+\alpha)\log n}(1+o(1)) \leq F_n \leq \frac{C_1 n}{(1+\alpha)\log n}(1+o(1)) \text{ w.h.p.}$$

The proof of Theorem 5.3 follows exactly the same steps as the proof of the corresponding main theorem (Theorem 4.13) from the previous chapter. In particular, we again make use of the *cut-point technique* that was already used in Chapter 3 and Chapter 4.

We start with the following lemma, which resembles Lemma 4.15 from Chapter 4.3. The key difference is the asymptotic behaviour of the number of fringe subtrees with k nodes as k increases: instead of a factor $k^{-3/2}$, we have a factor k^{-2}. Recall that the *shape* of a labeled tree denotes its underlying unlabeled tree.

Lemma 5.4. *Let T_n be a random tree of size n drawn from a very simple family of increasing trees with α defined as above. Let c, ε be positive real numbers with $\varepsilon < \frac{1}{2}$. For every positive integer k with $c \log n \leq k \leq n^\varepsilon$, let $\mathscr{S}_k$ be a subset of the possible shapes of a tree of size k, and let p_k be the probability that a random tree of size k from the given family has a shape that belongs to $\mathscr{S}_k$. Furthermore, let $X_{n,k}$ denote the (random) number of fringe subtrees of size k in the random tree T_n whose shape belongs to $\mathscr{S}_k$. Moreover, let $Y_{n,\varepsilon}$ denote the (random) number of arbitrary fringe subtrees of size greater than n^ε in T_n. Then*

(a) $\mathbb{E}(X_{n,k}) = \frac{np_k}{(1+\alpha)k^2}(1 + \mathcal{O}(1/k))$ *for all k with $c \log n \leq k \leq n^\varepsilon$,*

(b) $\mathbb{V}(X_{n,k}) = \mathcal{O}(p_k n/k^2)$ *for all k with $c \log n \leq k \leq n^\varepsilon$,*

(c) $\mathbb{E}(Y_{n,\varepsilon}) = \mathcal{O}(n^{1-\varepsilon})$, *and*

(d) with high probability, the following statements hold simultaneously:

(i) $|X_{n,k} - \mathbb{E}(X_{n,k})| \leq p_k^{1/2} k^{-1} n^{1/2+\varepsilon}$ *for all k with $c \log n \leq k \leq n^\varepsilon$,*

(ii) $Y_{n,\varepsilon} \leq n^{1-\varepsilon/2}$.

Proof. The proof is quite similar to the proof of Lemma 4.15. Let again $Z_{n,k}$ denote the number of fringe subtrees of size k in the random tree T_n. Again we find that $X_{n,k}$ can be regarded as a sum of $Z_{n,k}$ Bernoulli random variables with probability p_k. By [51, Theorem 15.1], we have

$$\mathbb{E}(X_{n,k}) = p_k \mathbb{E}(Z_{n,k})$$

as well as

$$\mathbb{V}(X_{n,k}) = p_k^2 \mathbb{V}(Z_{n,k}) + p_k(1 - p_k)\mathbb{E}(Z_{n,k}).$$

Now (a) and (b) both follow from Lemma 5.1. In order to estimate $\mathbb{E}(Y_{n,\varepsilon})$, observe that

$$\mathbb{E}(Y_{n,\varepsilon}) = \sum_{k > n^\varepsilon} \mathbb{E}(Z_{n,k}).$$

Now (c) also follows from Lemma 5.1. In order to show (d), we again apply Chebyshev's inequality:

$$\mathbb{P}\left(|X_{n,k} - \mathbb{E}(X_{n,k})| \le p_k^{1/2} n^{1/2+\varepsilon k^{-1}}\right) \le \frac{\mathbb{V}(X_{n,k})}{p_k k^{-2} n^{1+2\varepsilon}} = \mathcal{O}(n^{-2\varepsilon}).$$

Thus, by the union bound, the probability that the stated inequality fails for any k in the given range is only $\mathcal{O}(n^{-\varepsilon})$, such that the first statement holds w.h.p. Finally, by Markov's inequality, we find that

$$\mathbb{P}\left(Y_{n,\varepsilon} > n^{1-\varepsilon/2}\right) \le \frac{\mathbb{E}(Y_{n,\varepsilon})}{n^{1-\varepsilon/2}} = \mathcal{O}(n^{-\varepsilon/2}),$$

such that the second inequality holds w.h.p. as well. □

We are now able to prove Theorem 5.3. With Lemma 5.4 in mind, it is easy to see that the proof is completely analogous to the proof of Theorem 4.13. The only difference is that sums of the form $\sum_{a\le k\le b} k^{-3/2}$ become sums of the form $\sum_{a\le k\le b} k^{-2}$. Since the proof of Theorem 5.3 is line by line analogous to the proof of Theorem 4.13, we omit the details here in order to avoid repetitions.

5.4 d-ary fringe subtrees

Our first application of Theorem 5.3 is to count the number of distinct (shapes of) d-ary trees in a random d-ary increasing tree. We obtain the following theorem:

Theorem 5.5. *Let F_n be the number of distinct d-ary trees occurring among the fringe subtrees of a random d-ary increasing tree of size n. For the two constants*

$$\underline{c}(d) = \frac{d}{d-1}\log(d-1) + d^2 \sum_{k=1}^{\infty} \frac{\log k}{((d-1)k+d)((d-1)k+1)},$$
$$\overline{c}(d) = \frac{d}{d-1}\big(d\log d - (d-1)\log(d-1)\big)$$

the following holds:

(i) $\dfrac{\underline{c}(d)n}{\log n}(1+o(1)) \le \mathbb{E}(F_n) \le \dfrac{\overline{c}(d)n}{\log n}(1+o(1))$,

(ii) $\dfrac{\underline{c}(d)n}{\log n}(1+o(1)) \le F_n \le \dfrac{\overline{c}(d)n}{\log n}(1+o(1))$ *w.h.p.*

Proof. We verify that the conditions of Theorem 5.3 are satisfied. For this, we consider a partition into isomorphism classes, where we consider two trees as isomorphic, if their underlying unlabeled d-ary trees (i.e., their shapes) are identical. Note that (C1) holds in this case for the same reasons as before in the case of simply-generated families of trees: The family of (unlabeled) d-ary trees (see Example 4.2) can be modeled as a simply generated family of trees with weight generating series $\Phi(x) = (1+x)^d$. Using Theorem 4.6, we find that the constant C_1 from Theorem 5.3 evaluates in this case to $C_1 = d\log(d) - (d-1)\log(d-1)$.

We now verify (C2). For this, we have to take the number of increasing labelings into account. Recall from (5.1) that this number is given as

$$\frac{n!}{\prod_{v\in V(t)} |t[v]|}.$$

Let $f(t) = \log|t|$ denote a toll function, then the corresponding additive functional is given as

$$F(t) = \sum_{v\in V(t)} \log|t[v]|.$$

This quantity, i.e., the sum of the logarithms of all fringe subtree sizes, is also known as the *shape functional*, see [34]. We find that the additive functional F clearly satisfies the conditions of Lemma 5.2, with

$$\begin{aligned}\mu &= \sum_{k=1}^{\infty} \frac{(1-1/d)\log k}{((1-1/d)k+1)((1-1/d)k+1/d)} \\ &= d(d-1)\sum_{k=1}^{\infty} \frac{\log k}{((d-1)k+d)((d-1)k+1)}.\end{aligned}$$

In particular (as in the proofs of Theorem 4.20 and Theorem 4.25) we can now use Lemma 5.2 to show that there exists a sequence $(\varepsilon_k)_{k\geq 0}$ that tends to 0 as $k\to\infty$ with the property that

$$\mathbb{P}\Big(\prod_{v\in V(T_k)} |T_k[v]| \geq 2^{(\mu-\varepsilon_k)k}\Big) \geq 1-\varepsilon_k,$$

for every $k \geq 0$, where T_k denotes a random d-ary increasing tree of size k. We hence define the subset of isomorphism classes $\mathscr{J}_k \subseteq \mathscr{I}_k$ as the set of isomorphism classes, whose corresponding shape t satisfies

$$\prod_{v\in V(t)} |t[v]| \geq 2^{(\mu-\varepsilon_k)k}.$$

In particular, the probability that a random tree of size k belongs to a class in $\mathscr{J}_k$ is thus $1-o(1)$ as $k\to\infty$, such that condition (C2a) is satisfied.

In order to show that condition (C2b) holds as well, recall that the number of d-ary increasing trees with n nodes is precisely $\prod_{k=1}^{n-1}(1+k(d-1))$ (see Section 5.2), which can be written as (see e.g. [27, Lemma 6.5])

$$\prod_{k=1}^{n-1}(1+k(d-1)) \sim n!(d-1)^n \frac{n^{(2-d)/(d-1)}}{\Gamma(\frac{1}{d-1})},$$

where Γ denotes the gamma function. We find that for a given d-ary tree t with n nodes, the probability that a random increasing d-ary tree with n nodes has

the shape of t is

$$\frac{n!}{\prod_{k=1}^{n-1}(1+k(d-1))}\prod_{v\in V(t)}\frac{1}{|t[v]|}.$$

Note next that

$$\frac{n!}{\prod_{k=1}^{n-1}(1+k(d-1))}\sim\frac{\Gamma(\frac{1}{d-1})n^{(d-2)/(d-1)}}{(d-1)^n}=2^{-\log(d-1)n+o(\log n)}. \tag{5.2}$$

Thus, the probability that a random tree of size k lies in an isomorphism class in $\mathscr{J}_k$ is never greater than $2^{-(\log(d-1)+\mu)k+o(k)}$, such that condition (C2b) is satisfied, too. All in all, we find that condition (C2) holds as well with corresponding constant

$$C_2=\log(d-1)+d(d-1)\sum_{k=1}^{\infty}\frac{\log k}{((d-1)k+d)((d-1)k+1)}.$$

The statement now follows. □

In particular, for the special case $d=2$, which corresponds to binary search trees, we obtain the following corollary:

Corollary 5.6. *Let F_n be the total number of distinct fringe subtrees in a random binary search tree of size n. For two constants $\underline{c}(2)\approx 3.472754274$ and $\overline{c}(2)=4$, the following holds:*

(i) $\underline{c}(2)\dfrac{n}{\log n}(1+o(1))\le\mathbb{E}(F_n)\le\overline{c}(2)\dfrac{n}{\log n}(1+o(1))$,

(ii) $\underline{c}(2)\dfrac{n}{\log n}(1+o(1))\le F_n\le\overline{c}(2)\dfrac{n}{\log n}(1+o(1))$ *w.h.p.*

The upper bound in part (i) can already be found in [36] and [24]. Moreover, a lower bound of the form $\mathbb{E}(F_n)\ge cn/\log(n)(1+o(1))$ was already shown in [24] for the constant $c=(\log 3)/2\approx 0.7924812503$ and in [S8, S9] for the constant $c=0.8681$ (see also Example 3.29).

5.5 Plane fringe subtrees

As another application, we use Theorem 5.3 in this section in order to count the number of distinct plane fringe subtrees in random ports and random d-ary increasing trees. For random ports, we obtain the following result:

Theorem 5.7. *Let F_n be the number of distinct plane fringe subtrees in a random plane-oriented recursive tree of size n. For the constant*

$$c=\frac{1}{2}+\sum_{k=1}^{\infty}\frac{\log k}{(2k+1)(2k-1)}\approx 0.8446697909$$

the following holds:

(i) $\dfrac{cn}{\log n}(1+o(1)) \le \mathbb{E}(F_n) \le \dfrac{n}{\log n}(1+o(1))$,

(ii) $\dfrac{cn}{\log n}(1+o(1)) \le F_n \le \dfrac{n}{\log n}(1+o(1))$ *w.h.p.*

Proof. The procedure is analogous as in the proof of the previous theorem, i.e., we verify that the conditions of Theorem 5.3 are satisfied. We consider two trees as isomorphic, if their underlying unlabeled plane trees are identical. The isomorphism classes are precisely the plane trees, and we have

$$|\mathcal{I}_k| = \mathscr{C}_{k-1} = \frac{1}{k}\binom{2k-2}{k-1},$$

where $\mathscr{C}_{k-1}$ again denotes the $(k-1)^{st}$ Catalan number, thus $C_1 = \log 4 = 2$. In order to verify (C2), we argue in exactly the same way as before in the proof of Theorem 5.5. We again set $f(t) = \log|t|$ and find that the corresponding additive functional F satisfies the conditions of Lemma 5.2 with

$$\mu = \sum_{k=1}^{\infty} \frac{2\log k}{(2k+1)(2k-1)}.$$

Hence, by Lemma 5.2, there is a sequence $(\varepsilon_k)_{k\ge 0}$ that tends to 0 as $k \to \infty$ with the property that

$$\mathbb{P}\left(\prod_{v\in V(T_k)} |T_k[v]| \ge 2^{(\mu-\varepsilon_k)k}\right) \ge 1-\varepsilon_k,$$

for every $k \ge 0$, where T_k is a random port of size k. Thus, let $\mathscr{J}_k \subseteq \mathscr{I}_k$ be the set of isomorphism classes of increasing trees of size k, whose corresponding shape t satisfies

$$\prod_{v\in V(t)} |t[v]| \ge 2^{(\mu-\varepsilon_k)k}.$$

By Lemma 5.2, we find that condition (C2a) is satisfied. In order to show that condition (C2b) holds as well, we find that the number of ports of size n is precisely (see [27, Chapter 1.3.2])

$$\prod_{k=1}^{n-1}(2k-1) = \frac{1}{2^{n-1}}\frac{(2(n-1))!}{(n-1)!}.$$

We find that for a given plane tree t with n nodes, the probability that a random port with n nodes has the shape of t is

$$\frac{2^{n-1}n!(n-1)!}{(2(n-1))!}\prod_{v\in V(t)}\frac{1}{|t[v]|} \sim 2^{-n+o(n)}\prod_{v\in V(t)}\frac{1}{|t[v]|}.$$

Thus, the probability that a random tree of size k lies in an isomorphism class

in $\mathscr{J}_k$ is never greater than $2^{-(1+\mu)k+o(k)}$ and thus, condition (C2b) holds as well with

$$C_2 = 1 + \sum_{k\geq 1} \frac{2\log k}{(2k+1)(2k-1)}.$$

This proves the statement. □

Counting distinct plane fringe subtrees is also reasonable for the family of d-ary increasing trees. Let $\deg_r(t)$ denote the root degree of a tree t. We obtain the following result:

Theorem 5.8. *Let F_n be the number of distinct plane trees occurring among the fringe subtrees of a random d-ary increasing tree of size n. Let τ_d be the unique positive solution of the equation $1 = x^2 + 2x^3 + \cdots + (d-1)x^d$, and let*

$$\overline{c}(d) = \frac{d}{d-1}\log\left(1 + 2\tau_d + 3\tau_d^2 + \cdots + d\tau_d^{d-1}\right).$$

Furthermore, let

$$\underline{c}(d) = \frac{d}{d-1}\log(d-1) + d^2\sum_{k=1}^{\infty}\frac{\log k + \mathbb{E}(\log\binom{d}{\deg_r(T_k)})}{((d-1)k+d)((d-1)k+1)},$$

where T_k is a random d-ary increasing tree of size k. Then

(i) $\dfrac{\underline{c}(d)n}{\log n}(1+o(1)) \leq \mathbb{E}(F_n) \leq \dfrac{\overline{c}(d)n}{\log n}(1+o(1))$,

(ii) $\dfrac{\underline{c}(d)n}{\log n}(1+o(1)) \leq F_n \leq \dfrac{\overline{c}(d)n}{\log n}(1+o(1))$ *w.h.p.*

Proof. In this case, we consider fringe subtrees as distinct only if they are distinct as plane trees. Thus the isomorphism classes are plane trees with maximum degree at most d, which form a simply generated family of trees. The corresponding weight-sequence $(\phi_i)_{i\geq 0}$ satisfies $\phi_i = 1$ for $0 \leq i \leq d$ and $\phi_i = 0$ for $i > d$. The exponential growth constant of this simply generating family is $1 + 2\tau_d + 3\tau_d^2 + \cdots + d\tau_d^{d-1}$, see Theorem 4.6. Thus (C1) is satisfied with $C_1 = \log(1 + 2\tau_d + 3\tau_d^2 + \cdots + d\tau_d^{d-1})$. We also note that $2^{C_1} \in [3,4]$. Specifically, in the special case $d = 2$ we obtain the Motzkin numbers with $C_1 = \log 3$, see Example 4.3.

In order to verify (C2) and determine a suitable constant, we proceed analogously as in the proofs of the previous two theorems, except that we have to take the *weight* of a d-ary increasing tree into accout as well: The probability that a random increasing d-ary tree with n nodes has the shape of t, regarded as a plane tree, is

$$\frac{n!}{\prod_{k=1}^{n-1}(1+k(d-1))}\prod_{v\in V(t)}\frac{\binom{d}{\deg(v)}}{|t[v]|}.$$

Note here that the product $\prod_{v\in V(t)} \binom{d}{\deg(v)}$ gives the number of d-ary realizations of the plane tree t, see the proof of Theorem 4.20 for comparison. So we consider the additive functional with toll function

$$f(t) = \log|t| - \log\binom{d}{\deg_r(t)},$$

where $\deg_r(t)$ is the degree of the root of t. Since $\binom{d}{\deg_r(t)}$ is clearly bounded, the conditions of Lemma 5.2 are still satisfied. Let

$$\mu = \sum_{k=1}^{\infty} \frac{d(d-1)\mathbb{E}(f(T_k))}{((d-1)k+d)((d-1)k+1)},$$

where T_k is a random d-ary increasing tree of size k. Together with the estimate (5.2), we obtain a constant C_2 such that (C2) is satisfied as before, where

$$C_2 = \log(d-1) + \mu.$$

This proves the statement. □

In order to obtain the concrete value of the leading constant $\underline{c}(d)$ of the lower bound in Theorem 5.8, we have to determine the expectation $\mathbb{E}(\log\binom{d}{\deg_r(T_k)})$. For this, we need the probability that the root of a random d-ary increasing tree T_k of size k has degree i for $1 \le i \le d$. This is given in [7, Theorem 7]. For the case of binary trees, we obtain for example the following corollary from Theorem 5.8.

Corollary 5.9. *Let F_n be the number of distinct plane trees occurring among the fringe subtrees of a random binary increasing tree of size n. For the two constants $\overline{c}(2) = 2\log 3 \approx 3.1699250014$ and*

$$\underline{c}(2) = 4\sum_{k=2}^{\infty} \frac{\log k - \frac{2\log 2}{k}}{(k+1)(k+2)} = 4\sum_{k=2}^{\infty} \frac{\log k}{(k+1)(k+2)} - \frac{2}{3} \approx 2.8060876067$$

the following holds:

(i) $\dfrac{\underline{c}(2)n}{\log n}(1+o(1)) \le \mathbb{E}(F_n) \le \dfrac{\overline{c}(2)n}{\log n}(1+o(1)),$

(ii) $\dfrac{\underline{c}(2)n}{\log n}(1+o(1)) \le F_n \le \dfrac{\overline{c}(2)n}{\log n}(1+o(1))$ *w.h.p.*

5.6 Unordered fringe subtrees

In this section, two fringe subtrees are regarded as isomorphic, if they are identical as unordered unlabeled trees, that is, we partition the increasing trees of size k into isomorphism classes $\mathcal{I}_k$, where two trees are isomorphic if their corresponding unlabeled unordered representations are identical. This notion of

distinctness is reasonable for all three families of increasing trees considered in this chapter.

We again show that for this partition into isomorphism classes, the conditions of Theorem 5.3 are satisfied. Note that condition (C1) of Theorem 5.3 again holds in this case for the same reasons as in Section 4.6 on simply-generated families of trees, since we can apply Theorem 4.24. For the lower bound, in the same way as in the previous sections, we define a suitable additive functional. Here we also have to take the number of automorphisms into account, so there are now three factors that determine the probability that a random increasing tree in one of our very simple families is isomorphic to a fixed rooted unordered unlabeled tree t:

- the number of plane representations of t, which is given by (see Section 4.6)
$$\frac{\prod_{v\in V(t)} \deg(v)!}{|\operatorname{Aut} t|},$$
- the weight $\prod_{v\in V(t)} \phi_{\deg(v)}$, where $\phi_k = \binom{d}{k}$ for d-ary increasing trees, $\phi_k = \binom{r+k-1}{k}$ for gports, and $\phi_k = \frac{1}{k!}$ for recursive trees,
- the number of increasing labelings of any plane representation, which is
$$\frac{|t|!}{\prod_{v\in V(t)} |t[v]|}.$$

If we divide the product of all these three quantities by the number (more precisely: total weight) of n-node increasing trees in the specific family, we obtain the probability that a random increasing tree with $n = |t|$ nodes is isomorphic to t. So once again we consider a suitable additive functional that takes all these into account. For a tree t whose root degree is $\deg_r(t)$ and whose branches belong to k_t isomorphism classes with multiplicities $m_1, m_2, \dots, m_{k_t}$, we define the toll function

- for d-ary increasing trees by
$$f(t) = \log|t| + \log\bigl(m_1!m_2!\cdots m_{k_t}!\bigr) - \log\bigl(d(d-1)\cdots(d-\deg_r(t)+1)\bigr),$$
- for gports by
$$f(t) = \log|t| + \log\bigl(m_1!m_2!\cdots m_{k_t}!\bigr) - \log\bigl(r(r+1)\cdots(r+\deg_r(t)-1)\bigr),$$
- and for recursive trees by
$$f(t) = \log|t| + \log\bigl(m_1!m_2!\cdots m_{k_t}!\bigr).$$

We then obtain the following result.

Theorem 5.10. *Let F_n be the number of distinct unordered trees represented by the fringe subtrees of a random increasing tree T_n of size n from one of the very simple families of increasing trees. Let $M = \{i \in \mathbb{N}_0 \mid \phi_i > 0\}$ and let $C_1 = \log(b_M)$, where b_M is the constant in Theorem 4.24. Set*

$$\hat{\mu} = \sum_{k=1}^{\infty} \frac{(1+\alpha)\mathbb{E}(f(T_k))}{((1+\alpha)k+1)((1+\alpha)k-\alpha)}, \tag{5.3}$$

(where f is the toll function defined above) and

$$C_2 = \begin{cases} \log(d-1) + \log\hat{\mu} & \text{for } d\text{-ary increasing trees,} \\ \log(r+1) + \log\hat{\mu} & \text{for gports,} \\ \log\hat{\mu} & \text{for recursive trees.} \end{cases}$$

Then

(i) $$\frac{C_2 n}{(1+\alpha)\log n}(1+o(1)) \le \mathbb{E}(F_n) \le \frac{C_1 n}{(1+\alpha)\log n}(1+o(1)),$$

(ii) $$\frac{C_2 n}{(1+\alpha)\log n}(1+o(1)) \le F_n \le \frac{C_1 n}{(1+\alpha)\log n}(1+o(1)) \text{ w.h.p.}$$

Proof. It remains to show that conditions (C2a) and (C2b) are satisfied. Let F be the additive functional associated with f. Then the probability that a random tree with k nodes belongs to the same isomorphism class as a fixed k-node tree t is

$$2^{-F(t)} \cdot \begin{cases} \frac{k!}{\prod_{j=1}^{k-1}(1+(d-1)j)} & \text{for } d\text{-ary increasing trees,} \\ \frac{k!}{\prod_{j=1}^{k-1}((r+1)j-1)} & \text{for gports,} \\ k & \text{for recursive trees.} \end{cases}$$

We show that F satisfies the conditions of Lemma 5.2. The toll function $f(t)$ is $\mathcal{O}(\log|t| + \deg_r(t)\log(\deg_r(t))) = \mathcal{O}(\deg_r(t)\log|t|)$. So in order to show that the conditions of Lemma 5.2 are satisfied, one needs to bound the average root degree in a suitable way. For d-ary increasing trees, this is trivial. In the other two cases, this was done in [27]: In [27, p.243], it is shown that

$$\mathbb{E}(\deg_r(T_n)) = \log(n-1) + \mathcal{O}(1)$$

for a random recursive tree T_n of size n, and from [27, Theorem 6.11], we find that

$$\mathbb{E}(\deg_r(T_n)) \sim r\Gamma\left(\frac{r}{r+1}\right) n^{1/(r+1)}$$

for a random gport T_n of size n. Hence, we have $\mathbb{E}|f(T_n)| = \mathcal{O}(\log^2 n)$ and $\mathbb{E}|f(T_n)| = \mathcal{O}(n^{1/(r+1)}\log n)$ respectively in Lemma 5.2, which means that the conditions of that lemma are satisfied. We can conclude now as before that the

conditions of Theorem 5.3 hold. In particular, we have

$$\frac{k!}{\prod_{j=1}^{k-1}(1+(d-1)j)} \sim 2^{-\log(d-1)n+o(\log n)}$$

and

$$\frac{k!}{\prod_{j=1}^{k-1}((r+1)j-1)} \sim 2^{-\log(r+1)n+o(\log n)}$$

(see e.g. [27, Lemma 6.5 and p.252]), such that the constant C_2 attains the values as stated above. This proves the theorem. □

For recursive trees, the upper bound of $\mathcal{O}(n/\log n)$ was determined recently in [9]. The authors of that paper conjectured that this upper bound is asymptotically sharp and proved a lower bound of order $\sqrt{n}$. Indeed, our general theorem (Theorem 5.3) applies and confirms their conjecture.

In the case of recursive trees and ports, the constant C_1 is the logarithm of the growth constant for the number of unordered rooted trees (Pólya trees), see the proof of Corollary 4.27. In the case of binary increasing trees, the constant C_1 is given as the logarithm of the growth factor of the Wedderburn-Etherington numbers, see the proof of Corollary 4.26.

To determine the constant C_2 is more complicated: For this, we have to determine $\mathbb{E}(f(T_k))$, where T_k stands for a random increasing tree with k nodes. It seems difficult to determine the expected value $\mathbb{E}(f(T_k))$ exactly, and even numerical approximation is somewhat tricky (however, it is easy to compute simple lower bounds, as it is clear that $\mathbb{E}(f(T_k)) \geq \log k$).

For the cases of random recursive trees and random binary increasing trees, the leading constants in the lower bound were determined in [S11]. This leads to the following theorems:

Theorem 5.11. *Let F_n be the total number of distinct unordered trees represented by the fringe subtrees of a random recursive tree of size n. For two constants $\underline{c} \approx 1.3181041035$ and $\overline{c} \approx 1.5635317110$, the following holds:*

(i) $\dfrac{\underline{c}n}{\log n}(1+o(1)) \leq \mathbb{E}(F_n) \leq \dfrac{\overline{c}n}{\log n}(1+o(1))$,

(ii) $\dfrac{\underline{c}n}{\log n}(1+o(1)) \leq F_n \leq \dfrac{\overline{c}n}{\log n}(1+o(1))$ *w.h.p.*

Theorem 5.12. *Let F_n be the total number of distinct unordered trees represented by the fringe subtrees of a random binary search tree of size n. For two constants $\underline{c} \approx 2.2318529681$ and $\overline{c} \approx 2.6244631318$, the following holds:*

(i) $\dfrac{\underline{c}n}{\log n}(1+o(1)) \leq \mathbb{E}(F_n) \leq \dfrac{\overline{c}n}{\log n}(1+o(1))$,

(ii) $\dfrac{\underline{c}n}{\log n}(1+o(1)) \leq F_n \leq \dfrac{\overline{c}n}{\log n}(1+o(1))$ *w.h.p.*

For the proves of Theorem 5.11 and Theorem 5.12, we refer to the journal version [S11], respectively, the conference version [S10].

5.7 Conclusion and open problems

As in Chapter 4, we have obtained a quite general main theorem in this chapter, which covers many different notions of distinctness with respect to the question of estimating the number of distinct fringe subtrees in the random tree model of increasing trees. As the examples with explicit constants show, the upper and lower bounds they provide are again quite close.

A natural open question is again if it is possible to determine a concrete leading constant c (depending on the concrete notion of distinctness and the family of trees), instead of lower and upper bounds, such that the number F_n of distinct fringe subtrees in a random tree of size n asymptotically grows as $c \cdot n/\log n$ in expectation and with high probability.

In particular, for the binary seach tree model, the three independent approaches [36], [24] and [S10, S11] (see Corollary 5.6) with their different proof techniques all yield the asymptotic upper bound $\mathbb{E}(F_n) \leq 4n/\log n(1 + o(1))$ (with the same leading constant), so a very interesting question is whether this upper bound can be improved or whether it is possible to find a matching lower bound.

As in the case of simply generated families of trees, it is an open problem to determine the variance $\mathbb{V}(F_n)$, even for the well-studied binary search tree model and under the usual, non-generalized notion of distinctness.

Part II

Empirical Tree Entropy

Chapter 6

Entropy bounds for grammar-based tree compressors

6.1 Introduction

In this chapter, we focus on another aspect of tree compression: empirical entropy for trees. In particular, we propose a new notion of empirical entropy for trees, which we call *label-shape entropy*, as it incorporates both labels and shape of the tree and is able to capture possible dependencies between node labels and tree shape. Additionally, label-shape entropy works as a reasonable compressibility measure for unlabeled trees as well.

The families of trees we consider in this chapter are Σ-labeled full binary trees (Definition 2.7) and Σ-labeled (unranked) plane trees (Definition 2.8). Unlabeled trees will be identified with labeled trees over a unary alphabet, see Section 2.2.2. With σ we denote the size of Σ. The results presented in this chapter will be valid both for labeled and unlabeled trees. For technical reasons, in order to incorporate the unlabeled case, we set $\hat{\sigma} = \max\{\sigma, 2\}$.

As a main result of this chapter, we use our new notion of empirical entropy for trees in order to show an entropy bound for grammar-based tree compressors. This is motivated by recent results in the field of grammar-based string compression, where several upper bounds on the compression perfomance of grammar-based compressors in terms of higher order empirical entropy have been shown. Recall that a grammar-based compressor is an algorithm that produces for a given string s an SLP $\mathcal{G}_s$.

In order to upper-bound the compression performance of a grammar-based string compressor, the choice of a concrete binary encoding $B(\mathcal{G})$ of an SLP $\mathcal{G}$ is crucial. Kieffer and Yang [64] came up with such a binary encoding B and proved that under certain assumptions on the grammar-based compressor $s \mapsto \mathcal{G}_s$, the combined compressor $s \mapsto B(\mathcal{G}_s)$ yields a universal code with respect to the

family of finite-state information sources over finite alphabets. More precisely, it is needed that the size of the SLP $\mathcal{G}_s$ is bounded by $\mathcal{O}(|s|/\log_{\hat{\sigma}}|s|)$ where σ is the size of the underlying alphabet and $\hat{\sigma} = \max\{2, \sigma\}$. This upper bound is met by all grammar-based compressors that produce so-called irreducible SLPs [64], which is the case for e.g. LZ78 [107], Bisection [63], and Repair [72] after a small modification of the latter.

In their paper [89], Navarro and Ochoa used the binary encoding $B(\mathcal{G}_s)$ from [64] in order to prove for every word s over an alphabet of size σ the upper bound

$$|B(\mathcal{G}_s)| \leq |s| H_k(s) + o(|s| \log \hat{\sigma}) \tag{6.1}$$

for every $k \leq o(\log_{\hat{\sigma}}|s|)$. Here, $H_k(s)$ is the k^{th}-order empirical entropy of s, and the grammar-based compressor $s \mapsto \mathcal{G}_s$ must satisfy the upper bound $|\mathcal{G}_s| \leq \mathcal{O}(|s|/\log_{\hat{\sigma}}|s|)$. Similar but weaker upper bounds for more practical binary SLP-encodings have been shown in [45, 86].

The main goal of this chapter is to generalize the result (6.1) from strings to trees. This will be done in two steps: first, we show a corresponding result for the family of labeled full binary trees, which we then transfer to the family of labeled plane trees in a second step. In order to generalize the result (6.1) to trees, a generalization of grammar-based compression from strings to trees is needed, as well as a suitable notion of empirical entropy for trees.

In Section 6.3, we formally introduce *tree-straight line programs*, or shortly TSLPs. TSLPs are linear context-free tree grammars generating exactly one tree [15, 73]. They do not only generalize SLPs from strings to trees, but also, in terms of tree compression, represent a generalization of DAG compression (as considered in the first part of this work). Whereas DAGs only have the ability to share repeated fringe subtrees of a tree, TSLPs can also share repeated tree patterns with a "hole" (called contexts).

As in the case of string compression, in order to analyze the compression performance of grammar-based tree compressors, the choice of a concrete binary encoding $B(\mathcal{G})$ of an TSLP $\mathcal{G}$ is essential. For unlabeled full binary trees the results of Kieffer and Yang on universal grammar-based string compressors have been extended to trees in [105], [S3]. Whereas the universal tree encoder from [105] is based on DAGs (and needs a certain assumption on the average DAG size with respect to the input distribution), the encoder from [S3] uses TSLPs of size $\mathcal{O}(n/\log n)$. For this, a binary encoding of TSLPs similar to the one for SLPs from [64] is proposed. For our purposes, we extend the binary TSLP-encoding from [S3] to node-labeled full binary trees (which is straightforward).

In Section 6.4, we present our new notion of empirical entropy for trees called *label-shape entropy*, which we first define for labeled full binary trees (and generalize to labeled plane trees later in Section 6.6). Let us remark that in recent years, several notions of empirical tree entropy have been proposed with the aim of quantifying the compressibility of a given tree, notably label entropy [32, 33], degree entropy [60] and label-degree entropy and degree-label entropy, [46]. An overview over the notions of empirical entropy for trees, together with a systematic (theoretical and experimental) comparison of these entropy measures

will be presented in Chapter 7.

In Section 6.5, we prove our main result of this chapter (Theorem 6.21), which states that

$$|B(\mathcal{G}_t)| \leq H_k^{\ell s}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log \log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma, \tag{6.2}$$

where t is a full binary tree with n leaves, the grammar-based compressor $t \mapsto \mathcal{G}_t$ produces TSLPs of size $\mathcal{O}(n/\log_{\hat{\sigma}} n)$ for full binary trees of leafsize n with σ many node labels and $\hat{\sigma} = \max(2, \sigma)$. Moreover, $H_k^{\ell s}(t)$ is the k^{th}-order label-shape entropy of t and B is the extension of the binary TSLP-encoding described in [S3] from unlabeled full binary trees to labeled full binary trees. If $k \leq o(\log_{\hat{\sigma}} n)$ then this bound can be simplified to $|B(\mathcal{G}_t)| \leq H_k^{\ell s}(t) + o(n \log \hat{\sigma})$. The assumption $k \leq o(\log_{\hat{\sigma}} n)$ can also be found in the corresponding result for strings [89]. In fact, Gagie argued in [41] that the k^{th}-order empirical entropy for strings stops being a reasonable complexity measure for almost all strings of length n over alphabets of size σ when $k \geq \log_{\hat{\sigma}} n$. Our bound (6.2) can be seen as an extension of the bound (6.1) [89] from strings to trees.

In Section 6.6 we present a simple extension of our entropy notion and the entropy bound to labeled plane trees (Theorem 6.23). Moreover, we consider experimental results with real XML document trees showing that for these trees, the k^{th}-order label-shape entropy is indeed very small compared to the worst-case bit size. A labeled plane tree with n nodes and σ node labels can be encoded with $2n + \log(\sigma)n$ bits [48]. Up to low order terms, this is optimal. Table 6.1 shows the values of the k^{th}-order label-shape entropy (for $k = 1, 2, 4, 8$) divided by $2n + \log(\sigma)n$ for several real XML trees (that were also used in other experiments for XML compression, [74, 75]). For $k = 4$, these quotients never exceed 20% and for $k = 8$ all quotients are bounded by 13.5%.

The results of this chapter are published in [S6] (see also the conference version [S4]).

6.2 Shannon entropy and empirical distributions

We start off with some preliminary definitions and results related to Shannon entropy and empirical distributions, which will be needed later in the chapter.

Throughout the chapter, we make the convention that $0 \cdot \log(0) = 0$ and $0 \cdot \log(x/0) = 0$ for $x \geq 0$. Let A be a finite set and let $p\colon A \to [0, 1]$ be a probability mass function. We define the *Shannon entropy* of p as

$$H(p) = \sum_{a \in A} -p(a) \log p(a) = \sum_{a \in A} p(a) \log(1/p(a)).$$

(see also Section 3.6, where we considered the Shannon entropy in the context of random trees). We have $0 \leq H(p) \leq \log |A|$. A well-known generalization of Shannon's inequality states that for every probability mass function p on A and

any mapping $q\colon A \to [0,1]$ such that $\sum_{a\in A} q(a) \leq 1$ we have

$$H(p) = \sum_{a\in A} -p(a)\log p(a) \leq \sum_{a\in A} -p(a)\log q(a); \tag{6.3}$$

see [2] for a proof. Shannon's inequality is the special case where q is a probability mass function as well. The Kullback-Leibler divergence between two probability mass functions p, q on A (see [20, Section 2.3]) is defined as

$$D(p \,\|\, q) = \sum_{a\in A} p(a)\cdot\log(p(a)/q(a)). \tag{6.4}$$

It is known that $D(p \,\|\, q) \geq 0$ for all p, q (this follows from Shannon's inequality) and $D(p \,\|\, q) = 0$ if and only if $p = q$.

Let $\overline{a} = (a_1, a_2, \ldots, a_l)$ be a tuple of elements that are from some (not necessarily finite) set S. The *empirical distribution* $p_{\overline{a}}\colon \{a_1, a_2, \ldots, a_l\} \to [0,1]$ of $\overline{a}$ is defined by

$$p_{\overline{a}}(a) = \frac{|\{i \mid 1 \leq i \leq l,\ a_i = a\}|}{l}.$$

We use this (and the following) definition also for words over some alphabet by identifying a word $w = a_1 a_2 \cdots a_l$ with the tuple $(a_1, a_2, \ldots, a_l)$. The *unnormalized empirical entropy* of $\overline{a}$ is

$$H(\overline{a}) = l \cdot H(p_{\overline{a}}) = -\sum_{i=1}^{l} \log p_{\overline{a}}(a_i). \tag{6.5}$$

From (6.3) it follows that for a tuple $\overline{a} = (a_1, a_2, \ldots, a_l)$ with $a_1, \ldots, a_l \in S$ and real numbers $q(a) \geq 0$ ($a \in S$) with $\sum_{a\in\{a_1,\ldots,a_l\}} q(a) \leq 1$ we have

$$\sum_{i=1}^{l} -\log p_{\overline{a}}(a_i) \leq \sum_{i=1}^{l} -\log q(a_i). \tag{6.6}$$

6.3 Tree straight-line programs

In this section, we introduce tree straight-line programs (TSLPs) and use them for the compression of labeled full binary trees $t \in \mathcal{B}(\Sigma)$. Recall that full binary trees $t \in \mathcal{B}(\Sigma)$ can be considered as a terms over the alphabet Σ, see Definition 2.9. TSLPs can be viewed as a generalization of DAGs, as considered in the first part of this work. Recall that a tree t can be compressed into a (minimal) DAG by merging all occurrences of identical fringe subtrees into one copy of the respective fringe subtree. Formally, by identifying trees as terms, such a DAG can be represented by a set of rules of the form $A \to a(B, C)$ (respectively, $A \to a$). Here $a \in \Sigma$ is a node label of the full binary tree and A, B, C correspond to nodes of the DAG. The rule $A \to a(B, C)$ tells us that node A in the DAG corresponds to a fringe subtree of the full binary tree whose

root is labeled with a, and its left (respectively, right) outgoing edge leads to the fringe subtree corresponding to node B (respectively, C) in the DAG. The rule $A \to a$ says that node A represents a leaf labeled with a.

These rules formally define a *regular tree grammar* from which t (and only t) can be derived. The DAG nodes A, B, C etc. then become nonterminals of the regular tree grammar. There is a unique start nonterminal from which the whole tree t can be derived. For example, the minimal DAG of the full binary tree $t = a(b, a(b, a(b, a(b, b))))$ can be represented as a regular tree grammar with start nonterminal A_1 and with rules $A_1 \to a(b, A_2)$, $A_2 \to a(b, A_3)$, $A_3 \to a(b, A_4)$, $A_4 \to a(b, A_5)$, $A_5 \to b$.

The main limitation of DAGs for tree compression is that they only allow to exploit repetitions of *fringe* subtrees. Better compression results can be obtained by also exploiting repetitions of *non-fringe* subtrees. For this, the notion of a *context* is introduced in grammar-based tree compression: A context is a labeled full binary tree, where exactly one leaf is labeled with a special symbol $\circ \notin \Sigma$, which we call the *parameter*. Intuitively, this parameter node is a "hole", such that we can insert other trees and contexts into a context by replacing its special symbol $\circ$ with the tree or context that we want to insert (a formal definition follows). We denote the insertion of a full binary tree or context s into a context c as $c[s]$. Consider again the tree $t = a(b, a(b, a(b, a(b, b))))$. Let $c = a(b, \circ)$ denote a context, then t can be written as $t = c[c[c[c[b]]]]$, that is, the context c occurs four times in the full binary tree t. These four occurences of c cannot be shared by DAG-compression. However, they can be shared if we extend regular tree grammars by a second type of nonterminals that derive into contexts instead of trees. Then t can be derived using the rules $S \to C_1(C_1(b))$, $C_1 \to C_2(C_2)$, $C_2 \to a(b, \circ)$. Here, C_1 and C_2 are nonterminals of the second type, which derive into contexts, whereas the start nonterminal S is a nonterminal of the first type, which derives into a tree. These rules define a context-free tree grammar. If this grammar is acyclic, i.e., from a nonterminal we cannot reach the same nonterminal in an arbitrary number of derivation steps, and if every nonterminal A has exactly one rule with A on the left-hand side, then such a context-free grammar is called a TSLP (a formal definition follows).

In this thesis, as this is sufficient for our purposes, we will consider only a particular type of TSLPs which can be used for the compression of labeled full binary trees and which allows one parameter node per context. However, more general definitions of TSLPs exist which allow for example to compress ranked trees or which allow more than one parameter node per context. For a survey, see [73].

Formally, contexts and TSLPs are defined as follows in this work.

Contexts. A *context* c over the alphabet Σ is a Σ-labeled full binary tree, such that exactly one leaf is labeled with the special symbol $\circ \notin \Sigma$ (called the *parameter*), all other nodes are labeled with symbols from Σ. In the same way as Σ-labeled full binary trees can be identified with terms over the alphabet Σ (see Definition 2.9), we obtain a corresponding definition for the set of contexts:

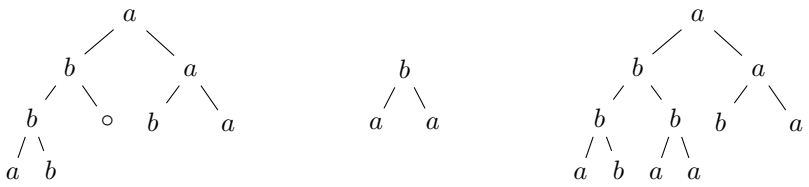


Figure 6.1: A context c (left), a tree t (middle), and the tree $c[t]$ (right).

Definition 6.1 (Contexts)**.** The set $\mathcal{C}(\Sigma)$ of *contexts* over the alphabet Σ is the smallest set of terms such that

- $\circ \in \mathcal{C}(\Sigma)$, and
- if $a \in \Sigma$, $c \in \mathcal{C}(\Sigma)$ and $t \in \mathcal{B}(\Sigma)$, then also $a(c,t), a(t,c) \in \mathcal{C}(\Sigma)$.

For a tree or context $t \in \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma)$ and a context $c \in \mathcal{C}(\Sigma)$, we denote by $c[t]$ the tree or context which results from c by replacing the unique occurrence of the parameter $\circ$ by t. Furthermore, for a context $c \in \mathcal{C}(\Sigma)$, we define its *leafsize* $\|c\|$ inductively by $\|\circ\| = 0$ and $\|a(c,t)\| = \|a(t,c)\| = \|t\| + \|c\|$ for $a \in \Sigma$, $c \in \mathcal{C}(\Sigma)$ and $t \in \mathcal{B}(\Sigma)$. In other words, $\|c\|$ is the number of leaves of c, where the unique occurrence of the parameter $\circ$ is not counted. Note that $\|c\| = \|c[a]\| - 1$ for every $a \in \Sigma$. We define $\mathcal{C}_n(\Sigma) = \{c \in \mathcal{C}(\Sigma) \mid \|c\| = n\}$ for $n \in \mathbb{N}$. Occasionally, we will consider a context as a graph with nodes and edges in the usual way, where each node is labeled with a symbol from $\Sigma \cup \{\circ\}$. Note that $c \in \mathcal{C}_n(\Sigma)$ has $2n+1$ nodes in total: $n+1$ leaves (including the parameter node) and n internal nodes.

Example 6.2. Let $\Sigma = \{a, b\}$. Then $c = a(b(b(a,b), \circ), a(b,a)) \in \mathcal{C}(\Sigma)$ is the context depicted on the left in Figure 6.1 and $t = b(a,a) \in \mathcal{B}(\Sigma)$ is the full binary tree depicted in the middle of Figure 6.1. The full binary tree $c[t] = a(b(b(a,b), b(a,a)), a(b,a)) \in \mathcal{B}(\Sigma)$ is shown on the right of Figure 6.1. The leafsize $\|c\|$ of the context c is $\|c\| = 4$.

Tree straight-line programs. Let V be a finite alphabet of symbols, where each symbol $A \in V$ has an associated rank 0 or 1 (also called a *ranked alphabet*). The elemens of V are called *nonterminals*. We assume that V contains at least one nonterminal of rank 0 and that V is disjoint from the set $\Sigma \cup \{\circ\}$ (the labels used for trees and contexts). We use V_0 (respectively, V_1) for the set of nonterminals of rank 0 (respectively, of rank 1). The idea is that nonterminals from V_0 (respectively, V_1) derive to trees from $\mathcal{B}(\Sigma)$ (respectively, contexts from $\mathcal{C}(\Sigma)$). We denote by $\mathcal{B}_V(\Sigma)$ the following set of ordered rooted trees over $\Sigma \cup V$: Each node of a tree $t \in \mathcal{B}_V(\Sigma)$ has zero, one or two children (and we do not distinguish between left-unary and right-unary nodes). Furthermore, each node is labeled with a symbol from $\Sigma \cup V$, such that nodes labeled by symbols from Σ have zero or two children and if a node is labeled with a symbol from V, then the number of children of this node corresponds to the rank of its label (a formal

Figure 6.2: A tree from $\mathcal{B}_V(\Sigma)$ (left) and a context from $\mathcal{C}_V(\Sigma)$ (right), where $a, b \in \Sigma$, $A \in V_0$ and $B \in V_1$.

definition follows). With $\mathcal{C}_V(\Sigma)$ we denote the corresponding set of all contexts, i.e., the set of all trees from $\mathcal{B}_V(\Sigma \cup \{\circ\})$, such that the parameter symbol $\circ$ occurs exactly once and at a leaf position. Formally, we define $\mathcal{B}_V(\Sigma)$ and $\mathcal{C}_V(\Sigma)$ as the smallest sets of formal expressions which satisfy the following conditions:

- $\Sigma \cup V_0 \subseteq \mathcal{B}_V(\Sigma)$ and $\circ \in \mathcal{C}_V(\Sigma)$,
- if $a \in \Sigma$, $A \in V_1$ and $t_1, t_2 \in \mathcal{B}_V(\Sigma)$, then $a(t_1, t_2), A(t_1) \in \mathcal{B}_V(\Sigma)$, and
- if $a \in \Sigma$, $A \in V_1$, $s \in \mathcal{C}_V(\Sigma)$ and $t \in \mathcal{B}_V(\Sigma)$, then $A(s) \in \mathcal{C}_V(\Sigma)$ and $a(s, t), a(t, s) \in \mathcal{C}_V(\Sigma)$.

Note that $\mathcal{B}(\Sigma) \subseteq \mathcal{B}_V(\Sigma)$ and $\mathcal{C}(\Sigma) \subseteq \mathcal{C}_V(\Sigma)$. A tree $t \in \mathcal{B}_V(\Sigma)$ and a context $c \in C_V(\Sigma)$ with $\Sigma = \{a, b\}$ and $V_0 = \{A\}$ and $V_1 = \{B\}$ are shown in Figure 6.2.

Definition 6.3. A *tree straight-line program* $\mathcal{G}$, or TSLP for short, is a tuple (V, A_0, r), where $r \colon V \to (\mathcal{B}_V(\Sigma) \cup \mathcal{C}_V(\Sigma))$ is a function which assigns to each nontermial its unique right-hand side and $A_0 \in V_0$ is the start nonterminal. Futhermore, if $A \in V_0$ (respectively, $A \in V_1$), then $r(A) \in \mathcal{B}_V(\Sigma)$ (respectively, $r(A) \in \mathcal{C}_V(\Sigma)$), and the binary relation $\{(A, B) \in V \times V \mid B \text{ occurs in } r(A)\}$ has to be acyclic.

These conditions ensure that exactly one tree is derived from the start nonterminal A_0 by using the rewrite rules $A \to r(A)$ for $A \in V$. To define this formally, we define $\mathsf{val}_\mathcal{G}(t) \in \mathcal{B}(\Sigma)$ for $t \in \mathcal{B}_V(\Sigma)$ and $\mathsf{val}_\mathcal{G}(c) \in \mathcal{C}(\Sigma)$ for $c \in \mathcal{C}_V(\Sigma)$ inductively by the following rules:

- $\mathsf{val}_\mathcal{G}(a) = a$ for $a \in \Sigma$ and $\mathsf{val}_\mathcal{G}(\circ) = \circ$,
- $\mathsf{val}_\mathcal{G}(a(s_1, s_2)) = a(\mathsf{val}_\mathcal{G}(s_1), \mathsf{val}_\mathcal{G}(s_2))$ for $a \in \Sigma$ and $s_1, s_2 \in \mathcal{B}_V(\Sigma) \cup \mathcal{C}_V(\Sigma)$ (and $s_1 \in \mathcal{B}_V(\Sigma)$ or $s_2 \in \mathcal{B}_V(\Sigma)$, since there is at most one parameter $\circ$ in $a(s_1, s_2)$),
- $\mathsf{val}_\mathcal{G}(A) = \mathsf{val}_\mathcal{G}(r(A))$ for $A \in V_0$,
- $\mathsf{val}_\mathcal{G}(A(s)) = \mathsf{val}_\mathcal{G}(r(A))[\mathsf{val}_\mathcal{G}(s)]$ for $A \in V_1$ and $s \in \mathcal{B}_V(\Sigma) \cup \mathcal{C}_V(\Sigma)$ (note that $\mathsf{val}_\mathcal{G}(r(A))$ is a context c, so $c[\mathsf{val}_\mathcal{G}(s)]$ is well-defined).

The tree defined by $\mathcal{G}$ is $\mathsf{val}(\mathcal{G}) = \mathsf{val}_\mathcal{G}(A_0) \in \mathcal{B}(\Sigma)$.

Example 6.4. Let $\Sigma = \{a, b\}$ and $\mathcal{G} = (\{A_0, A_1, A_2\}, A_0, r)$ be a TSLP such that $A_0, A_1 \in V_0$, $A_2 \in V_1$ and $r(A_0) = a(A_1, A_2(b))$, $r(A_1) = A_2(A_2(b))$, $r(A_2) = b(\circ, a)$. We have $\mathsf{val}_{\mathcal{G}}(A_2) = b(\circ, a)$, $\mathsf{val}_{\mathcal{G}}(A_1) = b(b(b, a), a)$ and $\mathsf{val}(\mathcal{G}) = \mathsf{val}_{\mathcal{G}}(A_0) = a(b(b(b, a), a), b(b, a))$.

Tree straight-line programs in normal form. For our purposes, it will be convenient to introduce TSLPs in a certain normal form: A TSLP $\mathcal{G} = (V, A_0, r)$ is in *normal form*, if the following conditions hold:

- $V = \{A_0, A_1 \ldots, A_{m-1}\}$ for some $m \in \mathbb{N}$, $m \geq 1$.
- For every $A_i \in V_0$, the right-hand side $r(A_i)$ is an expression of the form $A_j(\alpha)$, where $A_j \in V_1$ and $\alpha \in V_0 \cup \Sigma$.
- For every $A_i \in V_1$, the right-hand side $r(A_i)$ is an expression of the form $A_j(A_k(\circ))$, $a(\alpha, \circ)$, or $a(\circ, \alpha)$, where $A_j, A_k \in V_1$, $a \in \Sigma$ and $\alpha \in V_0 \cup \Sigma$.
- For every $A_i \in V$, define the word $\rho(A_i) \in (V \cup \Sigma)^*$ as follows:

$$\rho(A_i) = \begin{cases} A_j\alpha & \text{if } r(A_i) = A_j(\alpha) \\ A_jA_k & \text{if } r(A_i) = A_j(A_k(\circ)) \\ a\alpha & \text{if } r(A_i) = a(\alpha, \circ) \text{ or } a(\circ, \alpha). \end{cases}$$

  Let $\rho_{\mathcal{G}} = \rho(A_0)\rho(A_1)\cdots\rho(A_{m-1}) \in (\Sigma \cup \{A_1, A_2, \ldots, A_{m-1}\})^*$. Then we require that $\rho_{\mathcal{G}}$ is of the form $\rho_{\mathcal{G}} = A_1u_1A_2u_2\cdots A_{m-1}u_{m-1}$ with $u_i \in (\Sigma \cup \{A_1, A_2, \ldots, A_i\})^*$.
- $\mathsf{val}_{\mathcal{G}}(A_i) \neq \mathsf{val}_{\mathcal{G}}(A_j)$ for $i \neq j$.

We also allow the TSLP $\mathcal{G}_a = (\{A_0\}, A_0, A_0 \to a)$ for every $a \in \Sigma$ in order to obtain the singleton tree a. In this case, we set $\rho_{\mathcal{G}_a} = \rho(A_0) = a$.

Let $\mathcal{G} = (V, A_0, r)$ be a TSLP in normal form with $V = \{A_0, A_1, \ldots, A_{m-1}\}$ for the further definitions. We define the size of $\mathcal{G}$ as $|\mathcal{G}| = |V| = m$. Thus, $2|\mathcal{G}|$ is the length of $\rho_{\mathcal{G}}$. Let $\omega_{\mathcal{G}}$ be the word obtained from $\rho_{\mathcal{G}}$ by removing the first (i.e., left-most) occurrence of A_i from $\rho_{\mathcal{G}}$ for every $1 \leq i \leq m-1$. Thus, if $\rho_{\mathcal{G}} = A_1u_1A_2u_2\cdots A_{m-1}u_{m-1}$ with $u_i \in (\Sigma \cup \{A_1, A_2, \ldots, A_i\})^*$, then $\omega_{\mathcal{G}} = u_1u_2\cdots u_{m-1}$. Note that $|\omega_{\mathcal{G}}| = |\rho_{\mathcal{G}}| - m + 1 = m + 1$. The *entropy* $H(\mathcal{G})$ of the normal form TSLP $\mathcal{G}$ is defined as the empirical unnormalized entropy of the word $\omega_{\mathcal{G}}$ (see 6.5): $H(\mathcal{G}) = H(\omega_{\mathcal{G}})$.

Example 6.5. Let $\Sigma = \{a, b\}$ and $\mathcal{G} = (\{A_0, A_1, A_2, A_3, A_4\}, A_0, r)$ be the normal form TSLP with $A_0, A_2, A_3 \in V_0$, $A_1, A_4 \in V_1$ and

$$r(A_0) = A_1(A_2), r(A_1) = a(\circ, A_3), r(A_2) = A_4(A_3),$$
$$r(A_3) = A_4(b), r(A_4) = b(\circ, a).$$

We have $\mathsf{val}(\mathcal{G}) = a(b(b(b, a), a), b(b, a))$, $\rho_{\mathcal{G}} = A_1A_2aA_3A_4A_3A_4bba$ ($u_1 = \epsilon$, $u_3 = \epsilon$, $u_2 = a$, $u_4 = A_3A_4bba$), $|\mathcal{G}| = 5$ and $\omega_{\mathcal{G}} = aA_3A_4bba$.

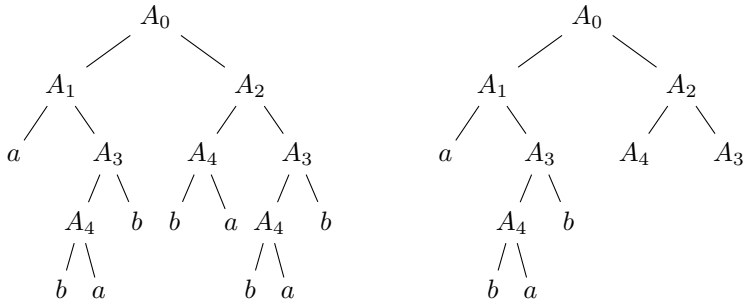


Figure 6.3: The derivation tree $\mathfrak{T}_{\mathcal{G}}$ of the TSLP from Example 6.6 (left) and an initial subtree $\mathfrak{T}'_{\mathcal{G}}$ of $\mathfrak{T}_{\mathcal{G}}$ (right).

The *derivation tree* $\mathfrak{T}_{\mathcal{G}}$ of the normal form TSLP $\mathcal{G}$ is a full binary tree with node labels from $V \cup \Sigma$. The root is labeled with A_0. Nodes labeled with a symbol from Σ are the leaves of $\mathfrak{T}_{\mathcal{G}}$. A node v that is labeled with a nonterminal A_i has $|\rho(A_i)| = 2$ many children. If $\rho(A_i) = \alpha\beta$ with $\alpha, \beta \in V \cup \Sigma$, then the left child of v is labeled with α and the right child is labeled with β. For every node u of $\mathfrak{T}_{\mathcal{G}}$ we define the tree or context $s_u = \mathsf{val}_{\mathcal{G}}(\alpha)$, where $\alpha \in V \cup \Sigma$ is the label of u. If $\alpha \in V_0 \cup \Sigma$, then $s_u \in \mathcal{B}(\Sigma)$ and if $\alpha \in V_1$ then $s_u \in \mathcal{C}(\Sigma)$. An *initial subtree* of the derivation tree $\mathfrak{T}_{\mathcal{G}}$ is a tree that can be obtained from $\mathfrak{T}_{\mathcal{G}}$ as follows: Take a subset U of the nodes of $\mathfrak{T}_{\mathcal{G}}$ and remove from $\mathfrak{T}_{\mathcal{G}}$ all proper descendants of nodes from U, i.e., all nodes that are located strictly below a node from U.

Example 6.6. Let $\mathcal{G}$ be the normal form TSLP from Example 6.5. The derivation tree $\mathfrak{T}_{\mathcal{G}}$ is shown in Figure 6.3 on the left; an initial subtree $\mathfrak{T}'_{\mathcal{G}}$ of $\mathfrak{T}_{\mathcal{G}}$ is shown on the right.

We obtain the following lemma with respect to initial subtrees of a derivation tree $\mathfrak{T}_{\mathcal{G}}$, which will be needed in order to prove our technical main lemma in this chapter (Lemma 6.19).

Lemma 6.7. *Let $\mathcal{G}$ be a TSLP in normal form with $t = \mathsf{val}(\mathcal{G})$. Let $\mathfrak{T}'$ be an initial subtree of the derivation tree $\mathfrak{T}_{\mathcal{G}}$, and let $v_1, \ldots, v_j$ be the sequence of all leaves of $\mathfrak{T}'$ (in left-to-right order). Then $2\|t\| \geq \sum_{i=1}^{j} \|s_{v_i}\|$.*

Proof. Let u be a node of $\mathfrak{T}_{\mathcal{G}}$ and let $\mathfrak{T}_u$ be the (fringe) subtree of $\mathfrak{T}_{\mathcal{G}}$ rooted in u. Then the nodes of s_u are in a one-to-one correspondence with the leaves of $\mathfrak{T}_u$, that is, if $s_u \in \mathcal{B}(\Sigma)$, we have $2\|s_u\| - 1 = \|\mathfrak{T}_u\|$, and if $s_u \in \mathcal{C}(\Sigma)$, we have $2\|s_u\| = \|\mathfrak{T}_u\|$ (recall that $\|\mathfrak{T}_u\|$ is the number of leaves of $\mathfrak{T}_u$). Thus, $2\|s_u\| - 1 \leq \|\mathfrak{T}_u\|$. Since $\mathfrak{T}'$ is an initial subtree of $\mathfrak{T}_{\mathcal{G}}$, we obtain in particular $2\|t\| - 1 = 2\|\mathsf{val}(\mathcal{G})\| - 1 = \|\mathfrak{T}_{\mathcal{G}}\| = \sum_{i=1}^{j} \|\mathfrak{T}_{v_i}\| \geq \sum_{i=1}^{j} (2\|s_{v_i}\| - 1)$. As $\|s_{v_i}\| \geq 1$, we have $2\|t\| \geq \sum_{i=1}^{j} 2\|s_{v_i}\| - j + 1 \geq \sum_{i=1}^{j} 2\|s_{v_i}\| + 1$ and the statement follows. □

A *grammar-based tree compressor* is an algorithm Ψ that produces for a given tree $t \in \mathcal{B}(\Sigma)$ a TSLP $\mathcal{G}_t$ in normal form such that $t = \mathsf{val}(\mathcal{G}_t)$. It is not hard

to show that every TSLP can be transformed with a linear size increase into a normal form TSLP that derives the same tree. For example, the TSLP from Example 6.4 is transformed into the normal form TSLP described in Example 6.5. We will not use this fact, since all we need is the following theorem from [42] (where $\hat{\sigma} = \max\{2, \sigma\}$):

Theorem 6.8 ([42])**.** *There exists a grammar-based compressor* Ψ *(working in linear time) with* $\max_{t \in \mathcal{B}_n(\Sigma)} |\mathcal{G}_t| \leq \mathcal{O}\left(n / \log_{\hat{\sigma}} n\right)$.

Binary encoding for TSLPs in normal form. Next, we specify a binary encoding for normal form TSLPs, which is a straightforward extension of the one for TSLPs producing unlabeled full binary trees from [S3] (which in turn is based on the encoding for SLPs from [64] and the encoding of DAGs from [105]). We only have to incorporate node labels into the encoding from [S3]. Let $\mathcal{G} = (V, A_0, r)$ be a TSLP in normal form with $m = |V| = |\mathcal{G}|$ nonterminals. We define the type $\mathsf{type}(A_i) \in \{0, 1, 2, 3\}$ of a nonterminal $A_i \in V$ as follows:

$$\mathsf{type}(A_i) = \begin{cases} 0 & \text{if } r(A_i) = A_j(\alpha) \text{ for some } A_j \in V_1, \alpha \in V_0 \cup \Sigma, \\ 1 & \text{if } r(A_i) = A_j(A_k(\circ)) \text{ for some } A_j, A_k \in V_1, \\ 2 & \text{if } r(A_i) = a(\alpha, \circ) \text{ for some } \alpha \in V_0 \cup \Sigma, a \in \Sigma, \\ 3 & \text{if } r(A_i) = a(\circ, \alpha) \text{ for some } \alpha \in V_0 \cup \Sigma, a \in \Sigma. \end{cases}$$

We define the binary word $B(\mathcal{G}) = w_0 w_1 w_2 w_3 w_4$, where the words $w_i \in \{0,1\}^+$ for $0 \leq i \leq 4$ are defined as follows:

- $w_0 = 0^{m-1}1$
- $w_1 = a_0 b_0 a_1 b_1 \cdots a_{m-1} b_{m-1}$, where $a_j b_j$ is the 2-bit binary encoding of $\mathsf{type}(A_j)$. Note that $|w_1| = 2m$.
- Let $\rho_{\mathcal{G}} = A_1 u_1 A_2 u_2 \cdots A_{m-1} u_{m-1}$ with $u_i \in (\Sigma \cup \{A_1, A_2, \ldots, A_i\})^*$. Then $w_2 = 10^{|u_1|}10^{|u_2|} \cdots 10^{|u_{m-1}|}$. Note that $|w_2| = 2m$.
- For $1 \leq i \leq m-1$ let $k_i = |\rho_{\mathcal{G}}|_{A_i} \geq 1$ be the number of occurrences of the nonterminal A_i in the word $\rho_{\mathcal{G}}$. Moreover, fix a total ordering on Σ. For $1 \leq i \leq \sigma$, let a_i denote the i^{th} symbol in Σ according to this ordering and let $l_i = |\rho_{\mathcal{G}}|_{a_i} \geq 0$ be the number of occurences of the symbol a_i in the word $\rho_{\mathcal{G}}$. Then $w_3 = 0^{k_1 - 1}10^{k_2 - 1}1 \cdots 0^{k_{m-1} - 1}10^{l_1}10^{l_2}1 \cdots 0^{l_\sigma}1$. Note that $|w_3| = 2m + \sigma$.
- The word w_4 encodes the word $\omega_{\mathcal{G}}$ using enumerative encoding [19]. Every nonterminal A_i, $1 \leq i \leq m-1$, has $\eta(A_i) := k_i - 1$ occurrences in $\omega_{\mathcal{G}}$. Every symbol $a_i \in \Sigma$, $1 \leq i \leq \sigma$, has $\eta(a_i) = l_i$ occurences in $\omega_{\mathcal{G}}$. Let S denote the set of words over the alphabet $\Sigma \cup \{A_1, \ldots, A_{m-1}\}$ with $\eta(a_i)$ occurrences of $a_i \in \Sigma$ $(1 \leq i \leq \sigma)$ and $\eta(A_i)$ occurrences of A_i $(1 \leq i \leq m-1)$. Hence,

$$|S| = \frac{(m+1)!}{\prod_{i=1}^{\sigma} \eta(a_i)! \prod_{i=1}^{m-1} \eta(A_i)!}. \tag{6.7}$$

Let $v_0, v_1, \ldots, v_{|S|-1}$ be the lexicographic enumeration of the words from S with respect to the alphabet order $a_1, \ldots, a_\sigma, A_1, \ldots, A_{m-1}$. Then w_4 is the binary encoding of the unique index i such that $\omega_\mathcal{G} = v_i$, where $|w_4| = \lceil \log_2 |S| \rceil$ (leading zeros are added to the binary encoding of i to obtain the length $\lceil \log_2 |S| \rceil$).

Example 6.9. Consider the normal form TSLP $\mathcal{G}$ from Example 6.5. We have $w_0 = 00001$, $w_1 = 0011000011$, $w_2 = 1101100000$ and $w_3 = 110101001001$. To compute w_4, note first that there are $|S| = 180$ words with two occurrences of a and b and one occurrence of A_3 and A_4. It follows that $|w_4| = \lceil \log(180) \rceil = 8$. Furthermore, with the canonical ordering on $\Sigma = \{a, b\}$, the order of the alphabet is a, b, A_3, A_4. The word $\omega_\mathcal{G} = aA_3A_4bba$ is the lexicographically largest word in S starting with aA_3. There are 132 words in S that are lexicographically larger than aA_3A_4bba, namely all words in S that start with b (60 words), A_3 (30 words), A_4 (30 words), or aA_4 (12 words). Hence $\omega_\mathcal{G} = aA_3A_4bba$ is the 48^{th} word in S in lexicographic order, i.e., $\omega_G = v_{47}$ and thus $w_4 = 00101111$.

The following lemma generalizes a result from [S3].

Lemma 6.10. *The set of code words $B(\mathcal{G})$, where $\mathcal{G}$ ranges over all TSLPs in a normal form, is a prefix code.*

Proof. Let $B(\mathcal{G}) = w_0w_1w_2w_3w_4$ with w_i defined as above. We show how to recover the TSLP $\mathcal{G}$, given the alphabet Σ and the ordering on Σ. From w_0 we can determine $m = |V|$ and the factors w_1, w_2 and w_3 of $B(\mathcal{G})$. Hence, we can determine the type of every nonterminal from w_1. The types allow to compute $\mathcal{G}$ from the word $\rho_\mathcal{G}$. Hence, it remains to determine $\rho_\mathcal{G}$. To compute $\rho_\mathcal{G}$ from w_2, one only needs $\omega_\mathcal{G}$. For this, one determines the frequencies $\eta(A_1), \ldots, \eta(A_{m-1}), \eta(a_1), \ldots, \eta(a_\sigma)$ of the symbols in $\omega_\mathcal{G}$ from w_3. Using these frequencies one computes the size $|S|$ from (6.7) and the length $\lceil \log |S| \rceil$ of w_4. From w_4, one can finally compute $\omega_\mathcal{G}$. □

Note that $|B(\mathcal{G})| \leq 7|\mathcal{G}| + \sigma + |w_4|$. By using the well-known bound on the code length of enumerative encoding [20, Theorem 11.1.3], we get the following bound, which extends [S3, Lemma 11] to node-labeled full binary trees:

Lemma 6.11. *For the length of the binary coding $B(\mathcal{G})$, we have*

$$|B(\mathcal{G})| \leq \mathcal{O}(|\mathcal{G}|) + \sigma + H(\mathcal{G}).$$

Intuitively, by Lemma 6.10, we can uniquely recover a full binary tree t from $B(\mathcal{G}_t)$, where $\mathcal{G}_t$ is a TSLP for t. By Lemma 6.11, we find that in order to upper-bound $|B(\mathcal{G}_t)|$ in terms of the k^{th}-order empirical tree entropy of t (plus lower-order terms), we can focus on the entropy $H(\mathcal{G}_t)$ of the grammar $\mathcal{G}_t$: Our main technical result in this chapter (Lemma 6.19) will provide a suitable upper bound on $H(\mathcal{G}_t)$. We also remark that a corresponding result (corresponding to Lemma 6.11) exists for strings ([64, Lemma 8] and [89, Lemma 2]).

6.4 Label-shape entropy for binary trees

In this section, we introduce our notion of empirical entropy of trees, which we call *label-shape entropy*, in order to distinguish it from other notions of empirical entropy for trees (see Chapter 7).

For the rest of this chapter, it will be convenient to identify a node v of a tree or context $s \in \mathcal{C}(\Sigma) \cup \mathcal{B}(\Sigma)$ with a bit string that describes the path from the root to the node (0 means left, 1 means right). More in detail, we define the set of nodes $V(s) \subseteq \{0,1\}^*$ of $s \in \mathcal{C}(\Sigma) \cup \mathcal{B}(\Sigma)$ as follows:

- $V(a) = \{\epsilon\}$ for $a \in \Sigma$,
- $V(\circ) = \emptyset$, and
- $V(a(s_0, s_1)) = \{0w \mid w \in V(s_0)\} \cup \{1w \mid w \in V(s_1)\} \cup \{\epsilon\}$ for every $a \in \Sigma$, $s_0, s_1 \in \mathcal{C}(\Sigma) \cup \mathcal{B}(\Sigma)$.

Throughout this chapter, a node v of a tree or context will be identified with the corresponding bit string.

Note that for a context $c \in \mathcal{C}(\Sigma)$, the set $V(c)$ does not contain the unique node in c labeled with the parameter $\circ$: This is an ad-hoc decision in order to increase the readability of this chapter, since for contexts, we mostly deal with the set of nodes without the parameter node. Also, it is possible to uniquely retrieve the path to the parameter $\circ$ in the context c from $V(c)$: For a labeled full binary tree $t \in \mathcal{B}(\Sigma)$, we have $w0 \in V(t)$ if and only if $w1 \in V(t)$ for all $w \in \{0,1\}^*$, as each node of t has zero or two children. The only context c which fulfills this property is $c = \circ$, as the parameter node is the only node of c and $V(c) = \emptyset$. For all other contexts $c \in \mathcal{C}(\Sigma)$, there exists a unique $w \in \{0,1\}^*$, such that $w0 \in V(c)$ (respectively, $w1 \in V(c)$) and $w1 \notin V(c)$ (respectively, $w0 \notin V(c)$). In this case, the parameter node is $w1$ (respectively, $w0$). Alternatively, the parameter node of a context c is the single node in the set $V(c[a]) \setminus V(c)$ for a symbol $a \in \Sigma$. We denote this node with $\omega(c) \in \{0,1\}^*$. In other words: $V(c[a]) \setminus V(c) = \{\omega(c)\}$. Consider a tree or context s and let $v \in V(s)$. The leaves of s are the strings in $V(s)$ that are maximal with respect to the prefix relation.

Example 6.12. Consider the tree $t = a(b(b(a,b),a),a(b,a))$ with $\Sigma = \{a,b\}$ depicted on the left of Figure 6.4. We have $V(t) = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001\}$. For the context $c = a(b(b(a,b),\circ),a(b,a))$ depicted on the right of Figure 6.4, we have $t = c[a]$ and $\omega(c) = 01$.

In the following, let $\hat{\lambda}_s \colon V(s) \to \Sigma \times \{0,2\}$ denote the function which maps a node v to the pair $(\lambda(v), \deg(v))$, that is, to the pair consisting of v's label $\lambda(v)$ and $v's$ degree $\deg(v)$. The mapping $\hat{\lambda}_s$ can be inductively defined as follows:

- $\hat{\lambda}_a(\epsilon) = (a,0)$ for $a \in \Sigma$,
- $\hat{\lambda}_s(\epsilon) = (a,2)$ for $s = a(s_0,s_1)$ with $a \in \Sigma$ and $s_0, s_1 \in \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma)$, and

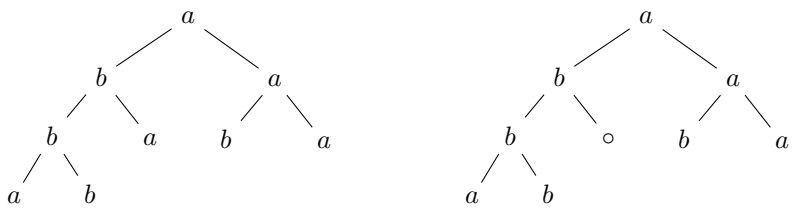


Figure 6.4: A tree (left) and a context (right).

- $\hat{\lambda}_s(iw) = \hat{\lambda}_{s_i}(w)$ for $s = a(s_0, s_1)$ with $a \in \Sigma$, $s_0, s_1 \in \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma)$ and $iw \in V(s)$.

Note that in the last case, if s is a context, we cannot have $s_i = \circ$, because we must have $w \in V(s_i)$. In the following, we will omit the subscript s in $\hat{\lambda}_s(v)$, if s is clear from the context.

Label-shape histories. In this paragraph, we introduce the crucial notion of the *label-shape history* of a node v in a tree or a context. Intuitively, the label-shape history of v records all the information that can be obtained by walking from the root of the tree/context straight down to the node v. This information consists of the directions (0 or 1) along the path from the root to v and the node labels along this path. First, we define the set of *label-shape histories* as

$$\mathcal{H} = (\Sigma\{0,1\})^* = \{a_1 i_1 \cdots a_n i_n \mid n \geq 0, a_k \in \Sigma, i_k \in \{0,1\} \text{ for all } 1 \leq k \leq n\}.$$

For an integer $k \geq 0$, let

$$\mathcal{H}_k = \{w \in \mathcal{H} \mid |w| = 2k\}$$

be the set of k-label-shape histories and let $\mathsf{suffix}_k \colon \mathcal{H} \to \mathcal{H}_k$ denote the partial function mapping a label-shape history $z \in \mathcal{H}$ with $|z| \geq 2k$ to the suffix of z of length $2k$, i.e., $\mathsf{suffix}_k(a_1 i_1 \cdots a_n i_n) = a_{n-k+1} i_{n-k+1} \cdots a_n i_n$ (the function suffix_0 maps a string to the empty string). For a tree t and a node $v \in V(t)$ (respectively, a context c and a node $v \in V(c) \cup \{\omega(c)\}$), we inductively define its label-shape history $h^{\ell s}(v) \in \mathcal{H}$ by

- $h^{\ell s}(\epsilon) = \epsilon$,
- $h^{\ell s}(wi) = h^{\ell s}(w)\lambda(w)i$ for $i \in \{0,1\}$, $wi \in V(t)$ (resp., $wi \in V(c) \cup \{\omega(c)\}$),

where $\lambda(w)$ again denotes the label of the node w. That is, in order to obtain $h^{\ell s}(v)$, while walking downwards in the tree from the root node to the node v, we alternately concatenate symbols from Σ with binary numbers in $\{0,1\}$ such that the symbol from Σ corresponds to the label of the current node and the binary number 0 (respectively, 1) states that we move on the the left (respectively, right) child node. Note that the label of v is not part of the label-shape history of v.

The k-label-shape history of a tree node $v \in V(t)$ is

$$h_k^{\ell s}(v) = \mathsf{suffix}_k((\Box 0)^k h^{\ell s}(v))$$

i.e., the suffix of length $2k$ of the string $(\Box 0)^k h^{\ell s}(v)$, where $\Box$ is a fixed dummy symbol in Σ (the choice is arbitrary). This means that if $|v| \geq k$, then $h_k^{\ell s}(v)$ describes the last k directions and node labels along the path from the root to node v. If $|v| < k$, we pad the label-shape history of v with $\Box$'s and zeros such that $h_k^{\ell s}(v) \in \mathcal{H}_k$. In general, there are several possible ways to define the k-label-shape history of a node whose label-shape history is shorter than $2k$. Reasonable approaches of how to deal with these nodes are

(i) to pad these label-shape histories with a fixed dummy symbol $\Box \in \Sigma$ and direction $i \in \{0, 1\}$,

(ii) to allow label-shape histories of length smaller than $2k$, or, equivalently, pad these label-shape histories with a fixed dummy symbol $\diamond \notin \Sigma$ and direction $i \in \{0, 1\}$, or

(iii) to ignore nodes whose label-shape history is of length smaller than $2k$.

In our definition, we use the variant (i) with $i = 0$. In [S6], it is shown that these choices only have a minor influence on our main results. The k-label-shape history is a natural extension of the k preceding symbols of a string position. For $z \in \mathcal{H}_k$, we denote with

$$V_z(t) = \{v \in V(t) \mid h_k^{\ell s}(v) = z\} \tag{6.8}$$

the set of nodes of t with k-label-shape history z.

Example 6.13. Consider the tree $t = a(b(b(a, b), a), a(b, a))$ from Figure 6.4 and let $\Box = a \in \Sigma$. Then, $h^{\ell s}(001) = h_3^{\ell s}(001) = a0b0b1$ and $h_4^{\ell s}(10) = a0a0a1a0$.

Label-shape entropy. We now come to the definition of our k^{th}-order empirical entropy for trees, which we call *label-shape entropy*, in order to distinguish it from existing concepts of empirical entropy for trees (see Chapter 7). As for strings, the term "empirical" refers to the fact that we assign an information content to a single tree instead of a probability distribution on trees. This has the advantage that empirical entropy is also useful in situations where we do not know the underlying probability distribution on trees. Let us fix $k \geq 0$ and let $t \in \mathcal{B}(\Sigma)$. For $z \in \mathcal{H}_k$, let

$$m_z^t = |V_z(t)| = |\{v \in V(t) \mid h_k^{\ell s}(v) = z\}| \tag{6.9}$$

be the number of nodes of t with k-label-shape history z and for $\tilde{a} \in \Sigma \times \{0, 2\}$ and $z \in \mathcal{H}_k$, let

$$m_{z,\tilde{a}}^t = |\{v \in V(t) \mid h_k^{\ell s}(v) = z \text{ and } \hat{\lambda}(v) = \tilde{a}\}|. \tag{6.10}$$

We define the k^{th}-order label-shape entropy as follows.

Definition 6.14 (Label-shape entropy for full binary trees)**.** Let $k \geq 0$. The k^{th}-order (unnormalized) label-shape entropy $H_k^{\ell s}(t)$ of a full binary tree $t \in \mathcal{B}(\Sigma)$ is defined as

$$H_k^{\ell s}(t) = \sum_{z \in \mathcal{H}_k} \sum_{\tilde{a} \in \Sigma \times \{0,2\}} m_{z,\tilde{a}}^t \log\left(\frac{m_z^t}{m_{z,\tilde{a}}^t}\right).$$

This definition extends the notion of k^{th}-order empirical entropy for strings to full binary trees. The k-label-shape history of a tree node takes the role of the k preceding characters of a string position. Note that

$$0 \leq H_k^{\ell s}(t) \leq (2n-1)\log(2\sigma) = (2n-1)(1+\log\sigma)$$

for every $t \in \mathcal{B}_n(\Sigma)$. This upper bound on the label-shape entropy matches the information theoretic bound for the worst-case output length of any tree encoder on $\mathcal{B}_n(\Sigma)$. Using the asymptotic bound (2.1) for the Catalan numbers, one sees that for any tree encoder there must exist a tree $t \in \mathcal{B}_n(\Sigma)$ which is encoded with $2\log(2\sigma)n - o(n) = 2(\log\sigma + 1)n - o(n)$ many bits. The k^{th}-order label-shape entropy $H_k^{\ell s}(t)$ is a lower bound on the coding length of a tree encoder that encodes for each node the relevant information (the label of the node and the binary information whether the node is a leaf or internal) depending on the k-label-shape history of the node: Thus, the k^{th}-order label-shape entropy is the expected uncertainty about the label and degree (0 or 2) of a node, given the last k directions and labels on the path from the root node to the node. Furthermore, note that the definition of label-shape entropy is also reasonable for unlabeled trees (i.e., trees over a unary alphabet): The k^{th}-order label-shape entropy of an unlabeled full binary tree tells us the expected uncertainty about the degree (0 or 2) of a node, given the last k directions on the path from the root node to the node.

Example 6.15. Let t denote the full binary tree $t = a(b(b(a,b),a),a(b,a))$ as depicted on the left of Figure 6.4. In order to compute the first-order label-shape entropy $H_1^{\ell s}(t)$ of t, we have to consider k-label-shape histories of t with $k = 1$: Let $\square = a$. We find $V_{a0}(t) = \{\epsilon, 0, 10\}$, $V_{b0}(t) = \{00, 000\}$, $V_{a1}(t) = \{1, 11\}$ and $V_{b1}(t) = \{01, 001\}$. Thus, we have $m_{a0}^t = 3$ and $m_{a1}^t = m_{b0}^t = m_{b1}^t = 2$. Next, for each k-label-shape history z, we consider $\hat{\lambda}(v)$ for $v \in V_z(t)$: For the label-shape history $z = a0$, we have $\hat{\lambda}(\epsilon) = (a,2)$, $\hat{\lambda}(0) = (b,2)$ and $\hat{\lambda}(10) = (b,0)$. Hence, we have $m_{a0,(a,2)}^t = m_{a0,(b,0)}^t = m_{a0,(b,2)}^t = 1$. Analogously, we obtain the values $m_{a1,(a,2)}^t = m_{a1,(a,0)}^t = 1$ as well as $m_{b0,(b,2)}^t = m_{b0,(a,0)}^t = 1$ and $m_{b1,(b,0)}^t = m_{b1,(a,0)}^t = 1$. Altogether, this yields $H_1^{\ell s}(t) = 3 \cdot \log(3) + 6 \cdot \log(2)$ which is roughly 10.75.

Label-shape processes. In the same way as k^{th}-order empirical entropy for strings corresponds to k^{th}-order Markov processes for strings [41], the label-shape entropy defined above corresponds to a particular kind of tree processes,

which we call label-shape processes. A *label-shape process* is an infinite tuple $\mathcal{P} = (P_z)_{z\in\mathcal{H}}$, where every P_z is a probability mass function on $\Sigma \times \{0,2\}$. A pair $(a,i) \in \Sigma \times \{0,2\}$ represents the information that a certain tree node v is labeled with a and has i children. The probability $P_z(a,i)$ is the probability that the tree node with label-shape history z is labeled with a and has i children. A label-shape process randomly generates a full binary tree $t \in \mathcal{B}(\Sigma)$ as follows: In a top-down way we determine for every tree node its label (from Σ) and its number of children, where this decision depends on the label-shape history of the tree node. We start at the root node, whose label-shape history is the empty word ϵ. If we reach a tree node v with label-shape history $z \in \mathcal{H}$, then we randomly choose a pair $(a,i) \in \Sigma \times \{0,2\}$ according to the probability mass function P_z. We assign the label $a \in \Sigma$ to v. If $i = 0$, then v becomes a leaf, otherwise the process continues at the two children $v0$ and $v1$ (whose label-shape history is well-defined). Note that in this way, we may produce infinite trees with non-zero probability. For finite trees $s \in \mathcal{B}(\Sigma)$, we obtain the probability

$$\mathsf{Prob}_{\mathcal{P}}(s) = \prod_{v\in V(s)} P_{h^{\ell_s}(v)}(\hat{\lambda}_s(v)). \tag{6.11}$$

For technical reasons, we will use this definition also for the case that s is a context. In other words: We associate with $\mathcal{P}$ the function $\mathsf{Prob}_{\mathcal{P}} \colon \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma) \to [0,1]$ using (6.11). Note that if s is a context, then the parameter node of s is not contained in $V(s)$ and therefore does not contribute to $\mathsf{Prob}_{\mathcal{P}}(s)$.

We first prove the following two lemmas, which will be needed in the proof of the main technical result from this chapter (Lemma 6.19). The reason that we obtain only an inequality in the following lemma is that the above tree generating process may also produce infinite trees with non-zero probability.

Lemma 6.16. *Let $\mathcal{P}$ be a label-shape process. Then $\sum_{t\in\mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}}(t) \leq 1$.*

Proof. First, note that as $\mathsf{Prob}_{\mathcal{P}}(t)$ is non-negative for every tree $t \in \mathcal{B}(\Sigma)$, the order of summation in the given sum does not matter: If $\sum_{t\in\mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}}(t) < \infty$, then this sum converges absolutely, and thus any rearrangement of the order of summation does not change its value. Define the set of trees $\mathcal{B}^{\leq n}$ inductively by $\mathcal{B}^{\leq 1} = \mathcal{B}_1(\Sigma)$ and

$$\mathcal{B}^{\leq n+1} = \mathcal{B}_1(\Sigma) \cup \{a(t_0,t_1) \mid a \in \Sigma, t_0, t_1 \in \mathcal{B}^{\leq n}\}.$$

Thus, $\mathcal{B}^{\leq n}$ is the set of all Σ-labeled full binary trees from $\mathcal{B}(\Sigma)$ of depth at most $n-1$. We find that $\mathcal{B}^{\leq n} \subsetneq \mathcal{B}^{\leq n+1}$ and $\mathcal{B}(\Sigma) = \bigcup_{n\geq 1} \mathcal{B}^{\leq n}$. It then suffices to show that $\sum_{t\in\mathcal{B}^{\leq n}} \mathsf{Prob}_{\mathcal{P}}(t) \leq 1$ for every $n \geq 1$. We prove the statement by induction on n. For this, it turns out to be useful to define for every $z \in \mathcal{H}$ the *shifted label-shape process* $\mathcal{P}_z = (P_{zz'})_{z'\in\mathcal{H}}$. We then prove by induction on n that $\sum_{t\in\mathcal{B}^{\leq n}} \mathsf{Prob}_{\mathcal{P}_z}(t) \leq 1$ for every $n \geq 1$ and all $z \in \mathcal{H}$ (in particular, for $z = \epsilon$ as well, which then yields the original statement). For $n = 1$, we have

$$\sum_{t\in\mathcal{B}^{\leq 1}} \mathsf{Prob}_{\mathcal{P}_z}(t) = \sum_{t\in\mathcal{B}_1(\Sigma)} \mathsf{Prob}_{\mathcal{P}_z}(t) = \sum_{a\in\Sigma} P_z(a,0) \leq 1.$$

Now assume that $n \geq 1$. We obtain

$$\begin{aligned}
&\sum_{t \in \mathcal{B}^{\leq n+1}} \mathsf{Prob}_{\mathcal{P}_z}(t) \\
&= \sum_{t \in \mathcal{B}^{\leq 1}} \mathsf{Prob}_{\mathcal{P}_z}(t) + \sum_{a \in \Sigma} \sum_{t_0 \in \mathcal{B}^{\leq n}} \sum_{t_1 \in \mathcal{B}^{\leq n}} \mathsf{Prob}_{\mathcal{P}_z}(a(t_0, t_1)) \\
&= \sum_{a \in \Sigma} P_z(a, 0) + \sum_{a \in \Sigma} \Big(P_z(a, 2) \cdot \sum_{t_0 \in \mathcal{B}^{\leq n}} \mathsf{Prob}_{\mathcal{P}_{za0}}(t_0) \cdot \sum_{t_1 \in \mathcal{B}^{\leq n}} \mathsf{Prob}_{\mathcal{P}_{za1}}(t_1) \Big) \\
&\leq \sum_{a \in \Sigma} P_z(a, 0) + \sum_{a \in \Sigma} P_z(a, 2) = 1.
\end{aligned}$$

This proves the lemma. □

Lemma 6.16 cannot be extended to contexts, but the following bound will suffice for our purpose:

Lemma 6.17. *Let $\mathcal{P}$ be a label-shape process. Then $\sum_{c \in \mathcal{C}_n(\Sigma)} \mathsf{Prob}_{\mathcal{P}}(c) \leq n + 1$ for every $n \geq 1$.*

Proof. In order to bound $\sum_{c \in \mathcal{C}_n(\Sigma)} \mathsf{Prob}_{\mathcal{P}}(c)$, we first represent the probability of each context $c \in \mathcal{C}_n(\Sigma)$ as a sum of probabilities of trees. Let $c \in \mathcal{C}_n(\Sigma)$ be a context. In order to bound $\mathsf{Prob}_{\mathcal{P}}(c)$, the idea is to consider the set $c[\mathcal{B}(\Sigma)] = \{c[t] \mid t \in \mathcal{B}(\Sigma)\}$ of all trees that arise from c by replacing the parameter by an arbitrary tree $t \in \mathcal{B}(\Sigma)$. As there might be infinite trees with positive probability with respect to $\mathcal{P}$, the sum of probabilities $\sum_{t \in c[\mathcal{B}(\Sigma)]} \mathsf{Prob}_{\mathcal{P}}(t)$ can be strictly smaller than $\mathsf{Prob}_{\mathcal{P}}(c)$. Thus, we fix an element $a \in \Sigma$ and modify $\mathcal{P}$ to a label-shape process $\mathcal{P}' = (P'_z)_{z \in \mathcal{H}}$, such that (i) $P'_z = P_z$ for $|z| \leq 2n$ and (ii) $P'_z(a, 0) = 1$ and $P'_z(a', i) = 0$ for every $(a', i) \in \Sigma \times \{0, 2\} \setminus \{(a, 0)\}$ and $|z| > 2n$. The label-shape process $\mathcal{P}'$ is created in such a way that all nodes v with $|v| \leq n$ contribute the probability $P_{h^{\ell s}(v)}(\hat{\lambda}(v))$ as before, and all other nodes are a-labeled leaves with probability 1. First, note that for each context $c \in \mathcal{C}_n(\Sigma)$ and each node $v \in V(c)$, we have $|v| \leq n$ and thus $P'_{h^{\ell s}(v)}(\hat{\lambda}(v)) = P_{h^{\ell s}(v)}(\hat{\lambda}(v))$. Secondly, all trees of depth larger than $n + 1$ have probability 0 with respect to $\mathcal{P}'$ (including infinite trees). Hence, we get $\sum_{t \in \mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}'}(t) = 1$. We obtain

$$\begin{aligned}
\sum_{t \in c[\mathcal{B}(\Sigma)]} \mathsf{Prob}_{\mathcal{P}'}(t) &= \sum_{t \in c[\mathcal{B}(\Sigma)]} \prod_{v \in V(t)} P'_{h^{\ell s}(v)}(\hat{\lambda}(v)) \\
&= \sum_{t \in c[\mathcal{B}(\Sigma)]} \prod_{v \in V(c)} P'_{h^{\ell s}(v)}(\hat{\lambda}(v)) \cdot \prod_{v \in V(t) \setminus V(c)} P'_{h^{\ell s}(v)}(\hat{\lambda}(v)) \\
&= \mathsf{Prob}_{\mathcal{P}}(c) \cdot \underbrace{\sum_{t \in c[\mathcal{B}(\Sigma)]} \prod_{v \in V(t) \setminus V(c)} P'_{h^{\ell s}(v)}(\hat{\lambda}(v))}_{(a)} .
\end{aligned}$$

We claim that (a) equals 1. To see this, consider the label-shape process $\mathcal{P}'' = (P''_z)_{z \in \mathcal{H}}$ with $P''_z = P'_{h^{\ell s}(\omega(c))z}$. For $\mathcal{P}''$, we find again that only finite

trees have non-zero probability and thus $\sum_{t\in\mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}''}(t) = 1$. We have

$$\begin{aligned}(a) &= \sum_{t\in\mathcal{B}(\Sigma)} \prod_{v\in V(t)} P'_{h^{\ell s}(\omega(c))h^{\ell s}(v)}(\hat{\lambda}(v)) \\ &= \sum_{t\in\mathcal{B}(\Sigma)} \prod_{v\in V(t)} P''_{h^{\ell s}(v)}(\hat{\lambda}(v)) \\ &= \sum_{t\in\mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}''}(t) = 1.\end{aligned}$$

It follows that $\mathsf{Prob}_{\mathcal{P}}(c) = \sum_{t\in c[\mathcal{B}(\Sigma)]} \mathsf{Prob}_{\mathcal{P}'}(t)$. In the second part of the proof it remains to bound

$$\sum_{c\in\mathcal{C}_n(\Sigma)} \mathsf{Prob}_{\mathcal{P}}(c) = \sum_{c\in\mathcal{C}_n(\Sigma)} \sum_{t\in c[\mathcal{B}(\Sigma)]} \mathsf{Prob}_{\mathcal{P}'}(t).$$

The key point here is that for each tree $t \in \mathcal{B}(\Sigma)$ there are at most $n+1$ different contexts $c \in \mathcal{C}_n(\Sigma)$, such that $t \in c[\mathcal{B}(\Sigma)]$. Note that for a tree t, the number of different contexts $c \in \mathcal{C}_n(\Sigma)$, such that $t \in c[\mathcal{B}(\Sigma)]$ is exactly the number of nodes $v \in V(t)$, such that replacing the fringe subtree rooted at v by the parameter $\circ$ yields a context c with leafsize $\|c\| = n$. This is the same as the number of fringe subtrees of t with $\|t\| - n$ leaves. Since different fringe subtrees in t of equal leafsize do not share nodes, we can bound the number of fringe subtrees with $\|t\| - n$ leaves by $\|t\|/(\|t\| - n)$. We can assume that $\|t\| = n + k$ for some $k > 0$ and the number of subtrees of t with $\|t\| - n$ leaves is at most $(n + k)/k = n/k + 1 \leq n + 1$. We obtain

$$\sum_{c\in\mathcal{C}_n(\Sigma)} \sum_{t\in c[\mathcal{B}(\Sigma)]} \mathsf{Prob}_{\mathcal{P}'}(t) \leq (n+1) \sum_{t\in\mathcal{B}(\Sigma)} \mathsf{Prob}_{\mathcal{P}'}(t) = 1.$$

This concludes the proof of the lemma. □

Label-shape processes of order k. A *k^{th}-order label-shape process* is a label-shape process $\mathcal{P} = (P_z)_{z\in\mathcal{H}}$ which satisfies the property that $P_z = P_{z'}$ if $\mathsf{suffix}_k((\Box 0)^k z) = \mathsf{suffix}_k((\Box 0)^k z')$. Thus, the probability mass function that is chosen for a certain tree node v depends only on the k-label-shape history of v. The set of k^{th}-order label-shape processes can be seen as a tree extension of k^{th}-order Markov processes for strings. We will identify the k^{th}-order label-shape process $\mathcal{P} = (P_z)_{z\in\mathcal{H}}$ with the finite tuple $(P_z)_{z\in\mathcal{H}_k}$ (recall that $\mathcal{H}_k$ is the set of k-label-shape histories); it contains all information about $\mathcal{P}$. Note that for a k^{th}-order label-shape process $\mathcal{P}$, we compute $\mathsf{Prob}_{\mathcal{P}}(s)$ for a tree or context s as

$$\mathsf{Prob}_{\mathcal{P}}(s) = \prod_{z\in\mathcal{H}_k} \prod_{v\in V_z(s)} P_z(\hat{\lambda}(v)), \tag{6.12}$$

where $V_z(s)$ is defined in (6.8) and the empty product is 1.

Let $t \in \mathcal{B}_n(\Sigma)$ be a Σ-labeled full binary tree. We define the *empirical k^{th}-order label-shape process* $\mathcal{P}^t = (P^t_z)_{z\in\mathcal{H}_k}$ corresponding to the full binary

tree t by

$$P_z^t(\tilde{a}) = \frac{m_{z,\tilde{a}}^t}{m_z^t} \tag{6.13}$$

for every $\tilde{a} \in \Sigma \times \{0,2\}$ and $z \in \mathcal{H}_k$ with $m_z^t > 0$, where m_z^t is defined in (6.9) and $m_{z,\tilde{a}}^t$ is defined in (6.10). Let $\tilde{a} = (a,i) \in \Sigma \times \{0,2\}$, then $P_z^t(\tilde{a})$ is the probability that a node of t that is randomly chosen among the nodes with label-shape history z is labeled with a and has i many children. If $m_z^t = 0$, then the choice for the probability mass function P_z^t is arbitrary. Note that the k^{th}-order label-shape entropy (as defined in Definition 6.14) of a Σ-labeled full binary tree t can be written as

$$H_k^{\ell s}(t) = \sum_{z \in \mathcal{H}_k} m_z^t \cdot H(P_z^t), \tag{6.14}$$

where $H(P_z^t)$ is the Shannon entropy of the probability mass function P_z^t (see Section 6.2).

The following theorem and its proof are very similar to a corresponding statement for strings shown by Gagie [41]. One obtains Gagie's result by replacing in the following theorem (i) k^{th}-order label-shape processes by k^{th}-order Markov processes and (ii) k^{th}-order label-shape entropy of trees by k^{th}-order empirical entropy of strings.

Theorem 6.18. *Let $t \in \mathcal{B}(\Sigma)$ be a Σ-labeled full binary tree. For every k^{th}-order label-shape process $\mathcal{P} = (P_z)_{z \in \mathcal{H}_k}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$, we have*

$$H_k^{\ell s}(t) \leq -\log \mathsf{Prob}_{\mathcal{P}}(t)$$

with equality if and only if $P_z^t = P_z$ for all $z \in \mathcal{H}_k$ with $m_z^t > 0$.

Proof. We have

$$\begin{aligned}
-\log \mathsf{Prob}_{\mathcal{P}}(t) &\overset{(6.12)}{=} \sum_{z \in \mathcal{H}_k} \sum_{v \in V_z(t)} \log(1/P_z(\hat{\lambda}(v))) \\
&\overset{(6.10)}{=} \sum_{z \in \mathcal{H}_k} \sum_{\tilde{a} \in \Sigma \times \{0,2\}} m_{z,\tilde{a}}^t \log(1/P_z(\tilde{a})) \\
&\overset{(6.13)}{=} \sum_{z \in \mathcal{H}_k} m_z^t \sum_{\tilde{a} \in \Sigma \times \{0,2\}} P_z^t(\tilde{a}) \cdot (\log(P_z^t(\tilde{a})/P_z(\tilde{a})) + \log(1/P_z^t(\tilde{a}))) \\
&\overset{(6.4)}{=} \sum_{z \in \mathcal{H}_k} m_z^t \cdot (D(P_z^t \,\|\, P_z) + H(P_z^t)) \\
&\overset{(6.14)}{\geq} H_k^{\ell s}(t)
\end{aligned}$$

with equality in the last line if and only if $P_z^t = P_z$ for all $z \in \mathcal{H}_k$ with $m_z^t > 0$. □

Theorem 6.18 will be a main ingredient in the proof of our main result (Theorem 6.21 in Section 6.5) in the same way as the above mentioned result from

[41] is used by Ochoa and Navarro [89] in order to bound the compression ratio of grammar-based string compressors in terms of the k^{th}-order empirical entropy of strings. In our main technical result (Theorem 6.20 in the following section), we show that for every k^{th}-order label-shape process $\mathcal{P}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$, we have that $|B(\mathcal{G}_t)|$ is upper-bounded by $-\log \mathsf{Prob}_{\mathcal{P}}(t)$ plus certain lower-order terms, where B is a binary TSLP-encoding and $t \to \mathcal{G}_t$ is a grammar-based tree compressor (as defined in Section 6.3). Our main result (Theorem 6.21), i.e., that $|B(\mathcal{G}_t)|$ is upper-bounded in terms of the k^{th} order label-shape entropy $H_k^{\ell s}(t)$ plus lower-order terms, will then easily follow as a corollary from Theorem 6.20 by applying Theorem 6.18.

6.5 Entropy bounds for binary encoded TSLPs

For this section we fix a grammar-based tree compressor $\Psi\colon t \to \mathcal{G}_t$, such that $\max_{t\in\mathcal{B}_n(\Sigma)} |\mathcal{G}_t| \leq \mathcal{O}\left(n/\log_{\hat{\sigma}} n\right)$ (where again $\hat{\sigma} = \max\{2,\sigma\}$); see Theorem 6.8. Let $\kappa > 0$ be a concrete constant such that

$$|\mathcal{G}_t| \leq \frac{\kappa n}{\log_{\hat{\sigma}} n} \tag{6.15}$$

for every tree $t \in \mathcal{B}_n(\Sigma)$ and n large enough. We allow that the alphabet size σ grows with n, i.e., $\sigma = \sigma(n)$ is a function in the tree size such that $1 \leq \sigma(n) \leq 2n-1$ (as a full binary tree $t \in \mathcal{B}_n(\Sigma)$ has $2n-1$ nodes). Our technical main result of this chapter is the following bound on the entropy $H(\mathcal{G}_t)$ (defined in Section 6.3). A similar bound (but for a different class of probability distributions on trees, which include, in particular, a class of leaf-centric binary tree sources as considered in Chapter 3) is stated in [S3, Lemma 13].

Lemma 6.19. *Let $k \geq 0$, $t \in \mathcal{B}_n(\Sigma)$ with $n \geq 2$ and let $\mathcal{P} = (P_w)_{w\in\mathcal{H}_k}$ be a k^{th}-order label-shape process with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. We have*

$$H(\mathcal{G}_t) \leq -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\left(\frac{kn\log\hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n\log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right).$$

Proof. Let $m = |\mathcal{G}_t| = |V|$ be the size of $\mathcal{G}_t$. Let $\mathfrak{T} = \mathfrak{T}_{\mathcal{G}_t}$ be the derivation tree of $\mathcal{G}_t$. We define an initial subtree $\mathfrak{T}'$ as follows: If v_1 and v_2 are non-leaf nodes of $\mathfrak{T}$ that are labeled with the same nonterminal and v_1 comes before v_2 in preorder (depth-first left-to-right), then we remove from $\mathfrak{T}$ all proper descendants of v_2. Thus, for every $A_i \in V$, there is exactly one non-leaf node in $\mathfrak{T}'$ that is labeled with A_i. For the TSLP from Example 6.5, the tree $\mathfrak{T}'$ is shown in Figure 6.3 on the right. We now use the tree $\mathfrak{T}'$ in order to define a factorization of the tree t into several subtrees, subcontexts and inner nodes. A similar factorization is also used in [S3, proof of Lemma 7].

Recall the definition of the words $\rho_{\mathcal{G}_t}$ and $\omega_{\mathcal{G}_t}$ from Section 6.3. A permutation of the word $\rho_{\mathcal{G}_t}$ is obtained by writing down for every node v of $\mathfrak{T}'$ the labels of $v's$ children (if they exist in $\mathfrak{T}'$) and then concatenating these labels. Moreover, the word $\omega_{\mathcal{G}_t}$ is obtained by writing down (in a suitable order) the labels of the leaves

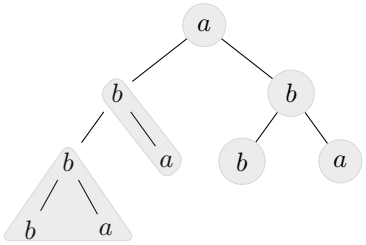


Figure 6.5: The tree $\mathsf{val}(\mathcal{G})$ of the TSLP from Example 6.5. The canonical occurrences of the trees/contexts or inner nodes used in the proof of Lemma 6.19 in $(a, b, a, b, \mathsf{val}_{\mathcal{G}}(A_4), \mathsf{val}_{\mathcal{G}}(A_3)) = (a, b, a, b, b(\circ, a), b(b, a))$ are highlighted.

of $\mathfrak{T}'$. Note that $\mathfrak{T}'$ has m non-leaf nodes and $m+1$ leaves. Let $v_1, v_2, \ldots, v_{m+1}$ be the sequence of all leaves of $\mathfrak{T}'$ (without loss of generality, in perorder) and let $\alpha_i \in \Sigma \cup \{A_1, \ldots, A_{m-1}\}$ be the label of v_i. Let $\bar{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{m+1})$. Then $\bar{\alpha}$ is a permutation of $\omega_{\mathcal{G}_t}$. We therefore have $|\omega_{\mathcal{G}_t}|_\alpha = |\bar{\alpha}|_\alpha$ for every $\alpha \in \Sigma \cup \{A_1, \ldots, A_{m-1}\}$. Hence, $p_{\bar{\alpha}}$ and $p_{\omega_{\mathcal{G}_t}}$ are the same empirical distributions. For example for the TSLP from Example 6.5, we find that $\bar{\alpha} = (a, b, a, b, A_4, A_3)$. Let $s_i = \mathsf{val}_{\mathcal{G}_t}(\alpha_i) \in \mathcal{B}(\Sigma) \cup (C(\Sigma) \setminus \{\circ\})$. Since $\mathsf{val}_{\mathcal{G}_t}(A_i) \neq \mathsf{val}_{\mathcal{G}_t}(A_j)$ for all $i \neq j$ (as $\mathcal{G}_t$ is in normal form) and $\mathsf{val}_{\mathcal{G}_t}(A_i) \notin \Sigma$ (this holds for every normal form TSLP that produces a tree of leafsize at least two), the tuple $\bar{s} = (s_1, s_2, \ldots, s_{m+1})$ satisfies for all $1 \leq i \leq m+1$:

$$p_{\omega_{\mathcal{G}_t}}(\alpha_i) = p_{\bar{s}}(s_i). \tag{6.16}$$

We define from $\mathcal{P}$ for every $z \in \mathcal{H}_k$ a label-shape process $\mathcal{P}_z = (P_{z,w})_{w \in \mathcal{H}}$ by setting

$$P_{z,w}(\tilde{a}) = P_{\mathsf{suffix}_k(zw)}(\tilde{a}) \tag{6.17}$$

for all $\tilde{a} \in \Sigma \times \{0, 2\}$. Note that the k^{th}-order label-shape process $\mathcal{P}$ is obtained for $z = (\square 0)^k$ for the fixed padding symbol $\square \in \Sigma$. We define a mapping $\vartheta \colon \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma) \to [0, 1]$ by

$$\vartheta(s) = \begin{cases} 1 & \text{if } s \in \mathcal{B}_1(\Sigma) = \Sigma \\ \max_{z \in \mathcal{H}_k} \mathsf{Prob}_{\mathcal{P}_z}(s) & \text{if } s \in (\mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma)) \setminus \mathcal{B}_1(\Sigma). \end{cases} \tag{6.18}$$

Thus, for every $s \in (\mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma)) \setminus \mathcal{B}_1(\Sigma)$, the function ϑ maximizes the values of the function $\mathsf{Prob}_{\mathcal{P}_z}$ associated with the k^{th}-order label-shape process $\mathcal{P}_z = (P_{z,w})_{w \in \mathcal{H}_k}$ by choosing an optimal prefix for the k-label-shape history of the nodes of s whose label-shape history is of length smaller than $2k$. We show that the mapping ϑ satisfies

$$\vartheta(t) \leq \prod_{i=1}^{m+1} \vartheta(s_i). \tag{6.19}$$

In order to prove (6.19), first note that by definition of the tree/context s_u, for each node u of the derivation tree $\mathfrak{T}$, the tree/context s_u corresponds to a subtree/subcontext or a single inner node of the tree t. We define a function χ which maps a node u of the derivation tree $\mathfrak{T}$ to a node $\chi(u) \in V(t) \subseteq \{0,1\}^*$. Intuitively, $\chi(u)$ is the root of the subtree/subcontext, respectively, the inner node of t which corresponds to s_u. Formally, χ is defined inductively as follows. For the root node u of $\mathfrak{T}$, we set $\chi(u) = \epsilon$. Furthermore, let u be a non-leaf node of $\mathfrak{T}$ which is labeled with the non-terminal A_i and for which $\chi(u)$ has been defined. Let u_1 be the left child and u_2 be the right child of u in $\mathfrak{T}$. We define $\chi(u_1) = \chi(u)$. The node $\chi(u_2)$ is defined as follows:

- If $r(A_i) = A_j(\alpha)$ with $A_j \in V_1$ and $\alpha \in V \cup \Sigma$, then $\chi(u_2) = \chi(u)\omega(s_{u_1})$ (recall that $\omega(s_{u_1}) \neq \epsilon$ is the position of the parameter $\circ$ in the context $s_{u_1} = \mathsf{val}_{\mathcal{G}}(A_j)$).
- If $r(A_i) = a(\alpha, \circ)$ (respectively, $r(A_i) = a(\circ, \alpha)$) for $a \in \Sigma$ and $\alpha \in \Sigma \cup V_0$, then we define $\chi(u_2) = \chi(u)0$ (respectively, $\chi(u_2) = \chi(u)1$).

This yields a well-defined function χ mapping a node u of $\mathfrak{T}$ to a node $\chi(u) \in V(t)$. Let us define

$$V_u(t) = \{\chi(u)v \mid v \in V(s_u)\} \subseteq V(t).$$

Then, the mapping

$$V(s_u) \ni v \mapsto \chi(u)v \in V_u(t) \tag{6.20}$$

is bijective. The definition of the sets $V_u(t)$ implies that if two nodes u and v of $\mathfrak{T}$ are not in an ancestor-descendant relationship, then $V_u(t) \cap V_v(t) = \emptyset$. Since the nodes $v_1, \ldots, v_{m+1}$ are the leaves of the initial subtree $\mathfrak{T}'$ and hence not in an ancestor-descendant relationship, the sets $V_{v_i}(t)$ are disjoint subsets of $V(t)$. For the TSLP from Example 6.5, the node sets $V_{v_1}(t), V_{v_2}(t), V_{v_3}(t), V_{v_4}(t), V_{v_5}(t)$ and $V_{v_6}(t)$ corresponding to the six leaves of the initial subtree depicted in Figure 6.3 (right) are shown in Figure 6.5. Note that if $s_i \notin \mathcal{B}_1(\Sigma)$, then the bijection from (6.20) also preserves the $\hat{\lambda}$-mapping in the following sense:

$$\hat{\lambda}_t(\chi(v_i)w) = \hat{\lambda}_{s_i}(w) \tag{6.21}$$

for every $w \in V(s_i)$. However, if $s_i \in \mathcal{B}_1(\Sigma)$ then this statement does not necessarily hold, as the number of children is not preserved in general: If $s_i \in \mathcal{B}_1(\Sigma)$, then s_i might correspond to a single inner node of t. In this case, we have $V_{v_i}(t) = \{\chi(v_i)\}$, $V(s_i) = \{\epsilon\}$ and $\hat{\lambda}_t(\chi(v_i)) = (a, 2)$ for some $a \in \Sigma$, but $\hat{\lambda}_{s_i}(\epsilon) = (a, 0)$. For example, in the TSLP from Example 6.5, the left-most leaf node of its initial subtree depicted in Figure 6.3 corresponds to the root node of the tree $\mathsf{val}(\mathcal{G})$ (see Figure 6.5). We define

$$\mathcal{I} := \{i \in \{1, \ldots, m+1\} \mid s_i \notin \mathcal{B}_1(\Sigma)\}.$$

In our running example, we have $(s_1, s_2, s_3, s_4, s_5, s_6) = (a, b, a, b, b(\circ, a), b(b, a))$

and hence $\mathcal{I} = \{5, 6\}$. The label-shape history $h^{\ell s}(\chi(v_i)w)$ of $\chi(v_i)w \in V_{v_i}(t)$ with $w \in V(s_i)$ in the tree t is the concatenation of the label-shape history $h^{\ell s}(\chi(v_i))$ of $\chi(v_i)$ in t and the label-shape history $h^{\ell s}(w)$ of w in the tree/context s_i. Thus, if $i \in \mathcal{I}$, we have

$$\begin{aligned}
&\max_{z\in\mathcal{H}_k} \prod_{v\in V_{v_i}(t)} P_{\mathsf{suffix}_k(zh^{\ell s}(v))}(\hat{\lambda}_t(v)) \\
= \;& \max_{z\in\mathcal{H}_k} \prod_{w\in V(s_i)} P_{\mathsf{suffix}_k(zh^{\ell s}(\chi(v_i))h^{\ell s}(w))}(\hat{\lambda}_t(\chi(v_i)w)) \\
\overset{(6.21)}{=} \;& \max_{z\in\mathcal{H}_k} \prod_{w\in V(s_i)} P_{\mathsf{suffix}_k(zh^{\ell s}(\chi(v_i))h^{\ell s}(w))}(\hat{\lambda}_{s_i}(w)) \\
\leq \;& \max_{z\in\mathcal{H}_k} \prod_{w\in V(s_i)} P_{\mathsf{suffix}_k(zh^{\ell s}(w))}(\hat{\lambda}_{s_i}(w)). \qquad (6.22)
\end{aligned}$$

The inequality in the last line follows from the fact that every k-label-shape history $\mathsf{suffix}_k(zh^{\ell s}(\chi(v_i))h^{\ell s}(w))$ for $z \in \mathcal{H}_k$ is also of the form $\mathsf{suffix}_k(z'h^{\ell s}(w))$ for some $z' \in \mathcal{H}_k$. We can now show (6.19). Since $t \in \mathcal{B}_n(\Sigma)$ with $n \geq 2$ we have

$$\begin{aligned}
\vartheta(t) &\overset{(6.18)}{=} \max_{z\in\mathcal{H}_k} \mathsf{Prob}_{\mathcal{P}_z}(t) \\
&\overset{(6.17)}{=} \max_{z\in\mathcal{H}_k} \prod_{v\in V(t)} P_{\mathsf{suffix}_k(zh^{\ell s}(v))}(\hat{\lambda}_t(v)) \\
&\overset{(*)}{\leq} \max_{z\in\mathcal{H}_k} \prod_{i\in\mathcal{I}} \prod_{v\in V_{v_i}(t)} P_{\mathsf{suffix}_k(zh^{\ell s}(v))}(\hat{\lambda}_t(v)) \\
&\leq \prod_{i\in\mathcal{I}} \max_{z\in\mathcal{H}_k} \prod_{v\in V_{v_i}(t)} P_{\mathsf{suffix}_k(zh^{\ell s}(v))}(\hat{\lambda}_t(v)) \\
&\overset{(6.22)}{\leq} \prod_{i\in\mathcal{I}} \max_{z\in\mathcal{H}_k} \prod_{w\in V(s_i)} P_{\mathsf{suffix}_k(zh^{\ell s}(w))}(\hat{\lambda}_{s_i}(w)) \\
&= \prod_{i=1}^{m+1} \vartheta(s_i) \quad (\text{since } \vartheta(s_i) = 1 \text{ for } i \notin \mathcal{I}),
\end{aligned}$$

where the inequality $(*)$ follows since $P_{\mathsf{suffix}_k(zh^{\ell s}(v))}(\hat{\lambda}(v)) \leq 1$ for $v \in V(t)$. Next, we define the function $\zeta\colon \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma) \setminus \{\circ\} \to [0, 1]$ as follows:

$$\zeta(s) = \begin{cases} 2^{-k-2}\sigma^{-k-1}\vartheta(s) & \text{if } s \in \mathcal{B}(\Sigma) \\ \dfrac{6}{\pi^2} \cdot 2^{-k-1}\sigma^{-k} \cdot \dfrac{\vartheta(s)}{\|s\|^2(\|s\|+1)} & \text{if } s \in \mathcal{C}(\Sigma) \setminus \{\circ\}. \end{cases}$$

The next step in our proof will be to find an upper bound for the sum $\sum_s \zeta(s)$ for $s \in \mathcal{B}(\Sigma) \cup \mathcal{C}(\Sigma) \setminus \{\circ\}$. For this, we first find that

$$\sum_{s\in\mathcal{B}(\Sigma)} \vartheta(s) \;=\; \sum_{s\in\mathcal{B}(\Sigma)\setminus\mathcal{B}_1(\Sigma)} \max_{z\in\mathcal{H}_k} \mathsf{Prob}_{\mathcal{P}_z}(s) + \sum_{s\in\mathcal{B}_1(\Sigma)} 1$$

$$\leq \sum_{z\in\mathcal{H}_k}\sum_{s\in\mathcal{B}(\Sigma)\setminus\mathcal{B}_1(\Sigma)} \mathsf{Prob}_{\mathcal{P}_z}(s) + \sigma \leq \ 2^k\sigma^k + \sigma, \tag{6.23}$$

where the last inequality follows from Lemma 6.16 and $|\mathcal{H}_k| = 2^k\sigma^k$. Similarly, we have

$$\begin{aligned}\sum_{s\in\mathcal{C}(\Sigma)\setminus\{\circ\}} \frac{\vartheta(s)}{\|s\|^2(\|s\|+1)} &= \sum_{r\geq 1}\frac{1}{r^2(r+1)}\sum_{s\in\mathcal{C}_r(\Sigma)}\max_{z\in\mathcal{H}_k}\mathsf{Prob}_{\mathcal{P}_z}(s)\\ &\leq \sum_{z\in\mathcal{H}_k}\sum_{r\geq 1}\frac{1}{r^2(r+1)}\sum_{s\in\mathcal{C}_r(\Sigma)}\mathsf{Prob}_{\mathcal{P}_z}(s)\\ &\leq 2^k\sigma^k\sum_{r\geq 1}\frac{1}{r^2} \leq \frac{\pi^2}{6}\cdot 2^k\sigma^k,\end{aligned} \tag{6.24}$$

where the last but one inequality follows from Lemma 6.17 and $|\mathcal{H}_k| = 2^k\sigma^k$, and the last inequality follows from the well-known fact that $\sum_{r\geq 1} r^{-2} = \pi^2/6$. With (6.23) and (6.24), we obtain

$$\begin{aligned}\sum_{s\in\mathcal{B}(\Sigma)\cup\mathcal{C}(\Sigma)\setminus\{\circ\}}\zeta(s) = \ & 2^{-k-2}\sigma^{-k-1}\sum_{s\in\mathcal{B}(\Sigma)}\vartheta(s) + \frac{6}{\pi^2}\cdot 2^{-k-1}\sigma^{-k}\sum_{s\in\mathcal{C}(\Sigma)\setminus\{\circ\}}\frac{\vartheta(s)}{\|s\|^2(\|s\|+1)}\\ \leq \ & 2^{-k-2}\sigma^{-k-1}\left(2^k\sigma^k+\sigma\right) + \frac{6}{\pi^2}\cdot 2^{-k-1}\sigma^{-k}\cdot\frac{\pi^2}{6}\cdot 2^k\sigma^k\\ = \ & 2^{-2}\sigma^{-1} + 2^{-k-2}\sigma^{-k} + \frac{1}{2}\\ \leq \ & 1.\end{aligned}$$

In particular, we have $\sum_{i=1}^m \zeta(s_i) \leq 1$. Thus, with Shannon's inequality (6.6), we obtain:

$$H(\mathcal{G}_t) = H(\omega_{\mathcal{G}_t}) = \sum_{i=1}^{m+1} -\log p_{\omega_{\mathcal{G}_t}}(\alpha_i) \overset{(6.16)}{=} \sum_{i=1}^{m+1} -\log p_{\overline{s}}(s_i) \leq \sum_{i=1}^{m+1} -\log\zeta(s_i).$$

With $\mathcal{I}_0 = \{i \mid 1\leq i\leq m+1, s_i\in\mathcal{B}(\Sigma)\}$ and $\mathcal{I}_1 = \{i \mid 1\leq i\leq m+1, s_i\in\mathcal{C}(\Sigma)\}$ we obtain

$$\begin{aligned}H(\mathcal{G}_t) \ &\leq \ -\sum_{i\in\mathcal{I}_0}\log\zeta(s_i) - \sum_{i\in\mathcal{I}_1}\log\zeta(s_i)\\ &= \ -\sum_{i\in\mathcal{I}_0}\log\left(2^{-k-2}\sigma^{-k-1}\vartheta(s_i)\right) - \sum_{i\in\mathcal{I}_1}\log\left(\frac{6}{\pi^2}\cdot\frac{2^{-k-1}\sigma^{-k}\vartheta(s_i)}{\|s_i\|^2(\|s_i\|+1)}\right)\end{aligned}$$

by definition of ζ. Using logarithmic identities, we obtain

$$\begin{aligned}H(\mathcal{G}_t) \leq |\mathcal{I}_0|\,(k+2+(k+1)\log\sigma) - \log\left(\prod_{i=1}^m\vartheta(s_i)\right) + \log\left(\frac{\pi^2}{6}\right)|\mathcal{I}_1|\\ + |\mathcal{I}_1|\,(k+1+k\log\sigma) + \sum_{i\in\mathcal{I}_1}\log\left(\|s_i\|^2(\|s_i\|+1)\right).\end{aligned}$$

Using $|\mathcal{I}_0|+|\mathcal{I}_1| = m+1 \leq 2m = 2|\mathcal{G}_t|$, $\log(\pi^2/6)|\mathcal{I}_1| \leq |\mathcal{I}_1|$ and $\|s_i\|+1 \leq 2\|s_i\|$, we obtain

$$H(\mathcal{G}_t) \leq 2(k+2)|\mathcal{G}_t| + 2(k+1)|\mathcal{G}_t| \log \sigma - \log\Big(\prod_{i=1}^{m+1} \vartheta(s_i)\Big) + \sum_{i=1}^{m+1} \log\left(2\|s_i\|^3\right).$$

Equation (6.19) and $\vartheta(t) \geq \mathsf{Prob}_{\mathcal{P}}(t)$ yield

$$\begin{aligned} H(\mathcal{G}_t) \;&\leq\; 2(k+3)|\mathcal{G}_t| + 2(k+1)|\mathcal{G}_t| \log \sigma - \log \vartheta(t) + 3\sum_{i=1}^{m+1} \log \|s_i\| \\ &\leq\; -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\bigg(k|\mathcal{G}_t| \log \hat{\sigma} + \sum_{i=1}^{m+1} \log \|s_i\|\bigg). \end{aligned}$$

Let us bound the sum $\sum_{i=1}^{m+1} \log \|s_i\|$: Using Jensen's inequality and Lemma 6.7 (which yields $\sum_{i=1}^{m+1} \|s_i\| \leq 2n$), we obtain

$$\begin{aligned} \sum_{i=1}^{m+1} \log \|s_i\| \;&\leq\; (m+1) \log\left(\sum_{i=1}^{m+1} \frac{\|s_i\|}{m+1}\right) \;\leq\; (m+1) \log\left(\frac{2n}{m+1}\right) \\ &\leq\; 2|\mathcal{G}_t| \log\left(\frac{2n}{|\mathcal{G}_t|}\right), \end{aligned}$$

and thus

$$H(\mathcal{G}_t) \leq -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\bigg(k|\mathcal{G}_t| \log \hat{\sigma} + |\mathcal{G}_t| \log\left(\frac{n}{|\mathcal{G}_t|}\right)\bigg). \tag{6.25}$$

In order to bound the term $|\mathcal{G}_t| \log(n/|\mathcal{G}_t|)$, recall that for n large enough we have $|\mathcal{G}_t| \leq \kappa \cdot n/\log_{\hat{\sigma}} n = \kappa \cdot n \cdot \log \hat{\sigma}/\log n$ by (6.15), where κ is a constant. Since $\sigma \leq 2n-1$ there is a constant $\kappa' \geq 1$ with $\kappa \cdot n/\log_{\hat{\sigma}} n \leq \kappa' n$. Since for every fixed $z \geq 1$, the function $f(x) = x \log\left(\frac{z}{x}\right)$ is monotonically increasing for $0 < x \leq \frac{z}{e}$ (where e is Euler's number), we get

$$|\mathcal{G}_t| \log\left(\frac{n}{|\mathcal{G}_t|}\right) \leq |\mathcal{G}_t| \log\left(\frac{e\kappa' n}{|\mathcal{G}_t|}\right) \leq \frac{\kappa n \log\left(\frac{e\kappa'}{\kappa} \log_{\hat{\sigma}} n\right)}{\log_{\hat{\sigma}} n} \leq \mathcal{O}\bigg(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\bigg).$$

With (6.25) we get

$$H(\mathcal{G}_t) \leq -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\bigg(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\bigg) + \mathcal{O}\bigg(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\bigg),$$

which proves the lemma. □

We consider the tree encoder $E_\Psi \colon \mathcal{B}(\Sigma) \to \{0,1\}^*$ defined by $E_\Psi(t) = B(\mathcal{G}_t)$ now, where as before the grammar-based tree compressor $\Psi \colon t \mapsto \mathcal{G}_t$ has to satisfy (6.15) for every $t \in \mathcal{B}_n(\Sigma)$ and n large enough. The following theorem says that the encoder E_Ψ is universal with respect to the class of all k^{th}-order label-shape processes. Universality means that the maximal pointwise redundancy [94]

converges to zero for $n \to \infty$.

Theorem 6.20. *For every $t \in \mathcal{B}_n(\Sigma)$, every $k \geq 0$ and every k^{th}-order label-shape process $\mathcal{P} = (P_z)_{z \in \mathcal{H}_k}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$, we have*

$$|E_\Psi(t)| \leq -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma.$$

Proof. Let $\mathcal{P} = (P_z)_{z \in \mathcal{H}_k}$ be a k^{th}-order label-shape process with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. Lemma 6.11 and Lemma 6.19 yield

$$\begin{aligned} |E_\Psi(t)| &\leq \mathcal{O}\left(|\mathcal{G}_t|\right) + H(\mathcal{G}_t) + \sigma \\ &\leq \mathcal{O}\left(|\mathcal{G}_t|\right) - \log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma \\ &= -\log \mathsf{Prob}_{\mathcal{P}}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma, \end{aligned}$$

where the last equality uses the bound $|\mathcal{G}_t| \leq \mathcal{O}\left(n / \log_{\hat{\sigma}} n\right)$. □

By taking in Theorem 6.20 for $\mathcal{P}$ the empirical k^{th}-order label-shape process $\mathcal{P}^t$ and using Theorem 6.18, we obtain our main result of this chapter:

Theorem 6.21. *For every $t \in \mathcal{B}_n(\Sigma)$ and every $k \geq 0$, we have*

$$|E_\Psi(t)| \leq H_k^{\ell s}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log\log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma.$$

6.6 Extension to labeled plane trees

So far, we have only introduced label-shape entropy for node-labeled full binary trees $t \in \mathcal{B}(\Sigma)$ (Definition 6.14). In this section, we consider an extension of label-shape entropy to Σ-labeled plane trees $t \in \mathcal{T}(\Sigma)$, that is, the number of children of a node can be any natural number and the children of every node are totally ordered, and each node is labeled by an element of a finite alphabet Σ. In particular, we show that the entropy bound for grammar-based tree compressors (Theorem 6.21) can be easily generalized to Σ-labeled plane trees. Recall the definition of the first-child next-sibling encoding (Definition 2.13), which transforms a Σ-labeled plane tree into a full binary tree. Using the first-child next-sibling encoding, we make the following definition.

Definition 6.22. We define the k^{th}-order label-shape entropy of a Σ-labeled plane tree $t \in \mathcal{T}(\Sigma)$ as $H_k^{\ell s}(t) = H_k^{\ell s}(\mathrm{fcns}(t))$.

Note that this definition is independent of the choice of $\square \in \Sigma$, which labels the newly added dummy-nodes in the first-child next-sibling encoding. From Theorem 6.21, we immediately obtain:

Theorem 6.23. *For every tree $t \in \mathcal{T}(\Sigma)$ with $|t| = n$ and every $k \geq 0$ we have*

$$|E_\Psi(\mathrm{fcns}(t))| \leq H_k^{\ell s}(t) + \mathcal{O}\left(\frac{kn \log \hat{\sigma}}{\log_{\hat{\sigma}} n}\right) + \mathcal{O}\left(\frac{n \log \log_{\hat{\sigma}} n}{\log_{\hat{\sigma}} n}\right) + \sigma.$$

Our definition of the k^{th}-order label-shape entropy of a Σ-labeled plane tree via the fcns-encoding has a practical motivation. Labeled plane trees occur for instance in the context of XML, where the hierarchical structure of a document is represented as a labeled plane tree. In this setting, the label of a node quite often depends on (i) the labels of the ancestor nodes and (ii) the labels of the (left) siblings. This dependence is captured by our definition of the k^{th}-order label-shape entropy. We also confirmed this intuition by experimental data (shown in Table 6.1) with real XML document trees (ignoring textual data at the leaves) showing that for these document trees, the k^{th}-order label-shape entropy is indeed very small compared to the worst-case bit size. More precisely, we computed for 21 XML document trees[1] the k^{th}-order label-shape entropy (for $k = 1, 2, 4, 8$) and divided the value by the worst-case bit length $2n + o(\log(\sigma)n)$, where n is the number of nodes and σ is the number of node labels [48].

Our experimental results combined with our entropy bound (Theorem 6.21) for grammar-based compression are in accordance with the fact that grammar-based tree compressors yield impressive compression ratios for XML document trees, see e.g. [74]. Some of the XML documents from our experiments were also used in [74], where the performance of the grammar-based tree compressor TreeRePair was tested. An interesting observation is that those XML trees, for which our k^{th}-order label-shape entropy is largest are indeed those XML trees with the worst compression ratio from TreeRePair [74]. This is in particular true for the Treebank document, see Table 6.1. TreeRePair obtained for Treebank a compression ratio of around 20%, whereas for all other documents tested in [74], TreeRePair achieved a compression ratio below 8%.

For the remainder of this section, it remains to remark the following. The above definition (Definition 6.22) of the k^{th}-order label-shape entropy of an Σ-labeled plane tree can be also applied to Σ-labeled full binary trees $t \in \mathcal{B}(\Sigma)$, as a labeled full binary tree can be considered as a labeled plane tree. This yields the question how the value $H_k^{\ell s}(t)$ (the label-shape entropy of t as defined in Definition 6.14 for full binary trees) relates to $H_k^{\ell s}(\mathrm{fcns}(t))$ (the label-shape entropy of t as defined in Definition 6.22 for labeled plane trees). For this, the following results are shown in [S6].

Lemma 6.24 ([S6], Lemma 8)**.** *Let $t \in \mathcal{B}(\Sigma)$ denote a full binary tree with first-child next-sibling encoding* $\mathrm{fcns}(t) \in \mathcal{B}(\Sigma)$*. Then $H_{2k}^{\ell s}(\mathrm{fcns}(t)) \leq H_{k-1}^{\ell s}(t)$ for $1 \leq k \leq \|t\|$.*

Lemma 6.25 ([S6])**.** *There is a sequence of full binary trees $(t_n)_{n \in \mathbb{N}}$, where $H_k^{\ell s}(\mathrm{fcns}(t_n))$ is exponentially smaller than $H_{k'}^{\ell s}(t_n)$ for every $n \geq 1$ and $k, k' \geq 2$ with $k, k' \leq o(n)$.*

[1] All data available from `http://xmlcompbench.sourceforge.net/Dataset.html`

XML document	n	σ	$w := (2 + \log \sigma)n$	H_1^{ls}/w	H_2^{ls}/w	H_4^{ls}/w	H_8^{ls}/w
Baseball	28 306	46	212 961.9447	2.9818 %	1.2547 %	0.6739 %	0.6662 %
DBLP	3 332 130	35	23 755 697.8193	10.9775 %	8.7407 %	8.2134 %	6.7270 %
DCSD-Normal	2 242 699	50	17 142 868.6330	4.2437 %	2.2481 %	1.7517 %	1.3038 %
EnWikiNew	404 652	20	2 558 180.8475	9.5317 %	3.0760 %	3.0759 %	2.9378 %
EnWikiQuote	262 955	20	1 662 382.6021	9.4270 %	3.1014 %	3.1014 %	3.1006 %
EnWikiVersity	495 839	20	3 134 658.5046	8.8952 %	2.3753 %	2.3753 %	2.3750 %
EXI-Array	226 523	47	1 711 288.1304	0.2506 %	0.2495 %	0.2492 %	0.2483 %
EXI-factbook	55 453	199	534 379.7451	2.2034 %	0.9450 %	0.8132 %	0.8092 %
EXI-Invoice	15 075	52	116 084.1288	0.0484 %	0.0268 %	0.0139 %	0.0098 %
EXI-Telecomp	177 634	39	1 294 135.1377	1.5405 %	0.0044 %	0.0034 %	0.0021 %
EXI-weblog	93 435	12	521 830.9713	0.0032 %	0.0028 %	0.0028 %	0.0028 %
Lineitem	1 022 976	18	6 311 685.1983	0.0003 %	0.0003 %	0.0003 %	0.0003 %
Mondial	22 423	23	146 277.8297	11.1285 %	9.2940 %	8.4702 %	7.7679 %
NASA	476 646	61	3 780 154.2290	7.7424 %	4.4588 %	3.8898 %	3.8054 %
Shakespeare	179 690	22	1 160 695.2676	11.9140 %	10.8416 %	10.6368 %	10.4765 %
SwissProt	2 977 031	85	25 035 017.5080	12.1892 %	10.5249 %	9.2455 %	8.1204 %
TCSD-Normal	2 749 751	24	18 107 007.2213	8.5450 %	8.4004 %	8.2862 %	8.2472 %
Treebank	2 437 666	250	24 293 253.5140	30.8912 %	23.0825 %	19.2444 %	13.4058 %
USHouse	6 712	43	49 845.0890	21.0500 %	18.2164 %	12.6572 %	9.3754 %
XMark1	167 865	74	1 378 079.8892	12.1610 %	9.5101 %	9.2271 %	8.4281 %
XMark2	1 666 315	74	13 679 535.2849	12.2125 %	9.5634 %	9.3259 %	8.9400 %

Table 6.1: Experimental results for XML tree structures, where n denotes the number of nodes and σ denotes the number of node labels.

6.7 Conclusion and open problems

We have proposed a new notion of empirical entropy for trees which incorporates node-labels as well as the shape of the tree and which is also suitable for the special case of unlabeled trees. Furthermore, we have shown an entropy bound for grammar-based tree compressors in terms of our new notion of empirical tree entropy, which generalizes a recent result [89] from string compression to tree compression.

Our definition of k^{th}-order label-shape entropy does not capture all regularities that can be exploited in grammar-based tree compression. Take for instance a complete unlabeled full binary tree t_n of leafsize 2^n (all paths from the root to a leaf have length n). This tree is well compressible: its minimal DAG has only $n+1$ nodes, hence there also exists a TSLP of size $n+1$ for t_n. But for every fixed k the k^{th}-order label-shape entropy of t_n divided by n converges to 2 (the trivial upper bound) for $n \to \infty$. If $n \gg k$ then for every k-label-shape history z the number of leaves with k-label-shape history z is roughly the same as the number of internal nodes with k-label-shape history z. Hence, although t_n is highly compressible with TSLPs (and even DAGs), its k^{th}-order label-shape entropy is close to the maximal value. However, the same situation also occurs for the other measures of empirical tree entropy proposed in [32, 46, 60] (as we will investigate in the next chapter).

Moreover, this phenomenon is also known for strings: in contrast to grammar-based compression, k^{th}-order empirical string entropy does not capture a particular kind of repetitiveness in strings. In [71, Lemma 2.6] it is shown that the unnormalized k^{th}-order empirical entropy of a string ww is at least twice the unnormalized k^{th}-order empirical entropy of w (as long as $k \leq |w|$). Furthermore, in [S6] it is shown that the gap between grammar-based compression and k^{th}-order empirical string entropy can be extreme in the following sense: there exists a sequence of strings S_n of length $\Theta(2^n)$ such that S_n has an SLP of size $\mathcal{O}(n)$ whereas the k^{th}-order empirial string entropy of S_n is of size $\Omega(2^{n-k})$ for $k \leq o(n)$.

For these reasons, several further compressibility measures for strings have been introduced throughout the years with the aim of capturing all kinds of regularities and types of repetitiveness in strings (see e.g. [62, 68], and see [85] for a survey). A basis for future research is to generalize these recent results on compressibility measures from strings to trees.

Another basis for future research is the following. Our tree representation based on tree straight-line programs can be queried in logarithmic time (if we assume logarithmic height of the grammar, which can be enforced by [43]). Currently open is to find a data structure for Σ-labeled trees which supports constant query time and which achieves an entropy bound in terms of the k^{th}-order label-shape entropy. For the special case of unlabeled plane trees, we will present such a compressed tree representation in Chapter 9.

Chapter 7

A comparison of empirical tree entropies

7.1 Introduction

Whereas in the area of string compression, there is basically only one notion of higher order empirical entropy (except for some minor modifications, as the k^{th}-order modified empirical entropy from [80]), the situation in the area of tree compression is different, as several notions of empirical tree entropy have been proposed in recent years. The main goal of this chapter is to give an overview and carry out a systematical comparison of these notions of empirical tree entropy. The notions of empirical entropy to be compared are the following (formal definitions will be given in Section 7.2).

Ferragina et al. [32, 33] introduced the k^{th}-order *label entropy* $H_k^\ell(t)$ of a labeled plane tree t. Its normalized version is the expected uncertainty about the label of a node v, given the *k-label history* of v which consists of the k first labels on the unique path from v's parent node to the root. Note that the k^{th}-order label entropy is not useful for unlabeled trees since it is mostly independent of the tree shape.

In [60], Jansson et al. introduce the *degree entropy* $H^{\deg}(t)$, which is the (unnormalized) 0^{th}-order empirical entropy of the node degrees occurring in the plane tree t. The degree entropy is mainly made for unlabeled trees since it ignores node labels. But in combination with label entropy it yields a reasonable measure for the compressibility of a tree: every node-labeled plane tree of size n in which σ different node labels occur can be stored in $H_k^\ell(t)+H^{\deg}(t)+o(n\log\hat{\sigma})$ bits, assuming that k and σ are not too big, where $\hat{\sigma}=\max\{2,\sigma\}$, see [46, 60].

Recently, Ganczorz [46] defined relativized versions of k^{th}-order label entropy and degree entropy: the k^{th}-order degree-label entropy $H_k^{\deg,\ell}(t)$ and the k^{th}-order label-degree entropy $H_k^{\ell,\deg}(t)$. The normalized version of $H_k^{\deg,\ell}(t)$ is the expected uncertainty about the label of a node v of t, given (i) the k-label-history of v and (ii) the degree of v, whereas the normalized version of

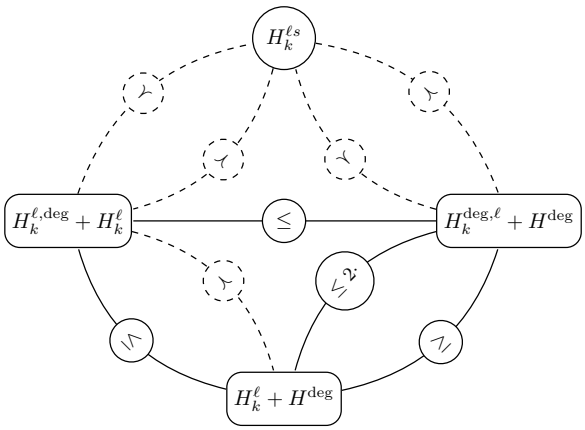


Figure 7.1: Theoretical comparison of the entropy bounds for Σ-labeled plane trees

$H_k^{\ell,\deg}(t)$ is the expected uncertainty about the degree of a node v, given (i) the k-label-history of v and (ii) the label of v. Ganczorz [46] proved that every Σ-labeled plane tree of size n can be stored in $H_k^{\ell}(t) + H_k^{\ell,\deg}(t) + o(n \log \hat{\sigma})$ bits as well as in $H^{\deg}(t) + H_k^{\deg,\ell}(t) + o(n \log \hat{\sigma})$ bits (again assuming that k and σ are not too big). Note that in the case of unlabeled trees t, we find that $H_k^{\ell}(t) + H_k^{\ell,\deg}(t) = H^{\deg}(t) + H_k^{\deg,\ell}(t) = H^{\deg}(t)$.

Moreover, in the previous chapter, we have introduced another notion of empirical entropy for trees, the *label-shape entropy* [S4, S6] (see Definition 6.14 and Definition 6.22). From Theorem 6.23, we know that a Σ-labeled plane tree t can be represented using at most $H_k^{\ell s}(t) + o(n \log \hat{\sigma})$ bits, under the assumption that k is not too big.

The goal of this chapter is to compare the entropy bounds $H_k^{\ell}(t) + H^{\deg}(t)$, $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$, $H^{\deg}(t) + H_k^{\deg,\ell}(t)$, and $H_k^{\ell s}(t)$. In Section 7.3, we start with a theoretical comparison of the entropy bounds. Our results for Σ-labeled plane trees (respectively, Σ-labeled full binary trees) are summarized in Figure 7.1 (respectively, Figure 7.2). Let us explain the meaning of the arrows in Figure 7.1 and Figure 7.2: For two entropy notions H and H', a dashed line from H to H' means that there is a sequence of Σ-labeled plane trees t_n ($n \geq 1$) such that (i) the function $n \mapsto |t_n|$ is strictly increasing and (ii) $H(t_n) \leq o(H'(t_n))$ (in most cases we prove an exponential separation). The meaning of a solid line from H to H' is that $H(t) \leq H'(t)$ for every Σ-labeled plane tree t. In particular, $H^{\deg}(t) + H_k^{\deg,\ell}(t)$ and $H^{\deg}(t) + H_k^{\ell}(t)$ are equivalent up to fixed multiplicative constants (which are 1 and 2). For the special case of Σ-labeled full binary trees, we find that the label-shape entropy $H_k^{\ell s}$ lower-bounds the other three entropy bounds.

We also investigate the relationship between the entropy bounds for unlabeled plane trees and unlabeled full binary trees. An unlabeled plane tree t of size n can be represented with $H^{\deg}(t) + o(n)$ bits [60]. We show the result that $H_k^{\ell s}(t) \leq 2H^{\deg}(t) + 2\log(n) + 4$ for every unlabeled plane tree t.

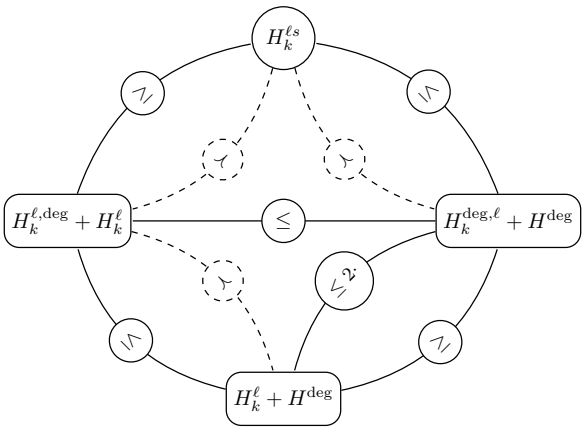


Figure 7.2: Theoretical comparison of the entropy bounds for Σ-labeled full binary trees

In Section 7.4, we underpin our theoretical investigations by experimental results with real XML data. For each XML document we consider the corresponding tree structure t (obtained by removing all text values and attributes) and compute $H_k^{\ell}(t) + H^{\deg}(t)$, $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$, $H^{\deg}(t) + H_k^{\deg,\ell}(t)$, and $H_k^{\ell s}(t)$. The results are summarized in Table 7.1, Table 7.2 and Table 7.3. Our experiments indicate that an entropy bound in terms of the label-shape entropy is the strongest for real XML data since the k^{th}-order label-shape entropy (for $k > 0$) is significantly smaller than all other values for all XML documents that we have examined. The results of this chapter are published in [S5].

7.2 Notions of empirical entropy for trees

In this chapter, we consider Σ-labeled full binary trees (Definition 2.7) and Σ-labeled plane trees (Definition 2.8). With σ we again denote the size of Σ and we set $\hat{\sigma} = \max\{\sigma, 2\}$. Unlabeled trees are identified with Σ-labeled trees over the unary alphabet $\Sigma = \{a\}$. For the entropy notions, we again make the convention that $0 \cdot \log 0 = 0$ and $0 \cdot \log(x/0) = 0$ for $x \geq 0$.

Recall the definition of the label-shape entropy $H_k^{\ell s}$ from the previous chapter (Definition 6.14 and Definition 6.22) and recall the definition of label-shape histories, see Section 6.4. We again make use of the notations

$$m_z^t = |\{v \in V(t) \mid h_k^{\ell s}(v) = z\}|$$
$$m_{z,\tilde{a}}^t = |\{v \in V(t) \mid h_k^{\ell s}(v) = z \text{ and } \hat{\lambda}(v) = \tilde{a}\}|,$$

as defined in (6.9) and (6.10), where $t \in \mathcal{B}(\Sigma)$ is a full binary tree, $z \in \mathcal{H}_k$ is a k-label-shape history, $\tilde{a} \in \Sigma \times \{0, 2\}$ and $\hat{\lambda}(v) = (\lambda(v), \deg(v))$ for nodes v of t. The concept of label histories is quite similar to the notion of label-shape histories.

Label histories. For a node $v \in V(t)$ of a tree $t \in \mathcal{T}(\Sigma)$, we define its label history $h^\ell(v) \in \Sigma^*$ as follows. For the root node v of t, we set $h^\ell(v) = \epsilon$ (i.e., the empty string), and for a child node w of a node v of t, we set $h^\ell(w) = h^\ell(v)\lambda(v)$. In other words, $h^\ell(v)$ is obtained by concatenating the node labels along the unique path from the root to v. Note that the label of v is not part of the label history of v. The k-label history $h^\ell_k(v)$ of a tree node $v \in V(t)$ is defined as the length-k-suffix of $\Box^k h^\ell(v)$, where $\Box$ is again a fixed dummy symbol in Σ. This means that if v has at least k ancestors in t, then $h^\ell_k(v)$ describes the last k node labels along the path from the root node to node v. Otherwise, we pad its label history $h^\ell(v)$ with the symbol $\Box \in \Sigma$, such that $h^\ell_k(v) \in \Sigma^k$. In general, there are again several ways how to treat nodes of depth smaller than $k+1$ in the definition of k-label histories. Here, the same alternatives are possible as for label-shape histories, see Section 6.4, and with the same arguments, it can be shown that these choices only have a minor influence on the entropy notions and the corresponding results (see [S6, Theorem 7]).

For $t \in \mathcal{T}(\Sigma)$, $z \in \Sigma^k, a \in \Sigma$ and $i \in \mathbb{N}_0$, we set

$$n^t_z = |\{v \in V(t) \mid h^\ell_k(v) = z\}|, \tag{7.1}$$

$$n^t_{z,a} = |\{v \in V(t) \mid h^\ell_k(v) = z \text{ and } \lambda(v) = a\}|, \tag{7.2}$$

$$n^t_i = |\{v \in V(t) \mid \deg(v) = i\}|, \tag{7.3}$$

$$n^t_{z,i} = |\{v \in V(t) \mid h^\ell_k(v) = z \text{ and } \deg(v) = i\}|, \tag{7.4}$$

$$n^t_{z,i,a} = |\{v \in V(t) \mid h^\ell_k(v) = z,\ \lambda(v) = a \text{ and } \deg(v) = i\}|. \tag{7.5}$$

In order to avoid ambiguities in these notations we should assume that $\Sigma \cap \mathbb{N}_0 = \emptyset$. Moreover, when writing $n^t_{z,i}$ (resp., $n^t_{z,a}$) then, implicitly, i (resp., a) always belongs to $\mathbb{N}_0$ (resp., Σ). Note that whereas label-shape histories are only defined for the special case of labeled full binary trees $t \in \mathcal{B}(\Sigma)$, label-histories are defined for labeled plane trees $t \in \mathcal{T}(\Sigma)$. We now formally define the various entropy measures that were mentioned in the introductory section of this chapter. In all cases we define *unnormalized entropies*, which has the advantage that we do not have to multiply with the size of the tree in bounds for the encoding size of a tree. In [32, 46, 60], the authors define normalized entropies, which are obtained by dividing the unnormalized entropy by the tree size.

Label entropy. The first notion of empirical entropy for trees was introduced in [32]. In order to distinguish the notions, we call the empirical entropy from [32] *label entropy*. It is defined for Σ-labeled plane trees $t \in \mathcal{T}(\Sigma)$.

Definition 7.1 (Label entropy, [32])**.** The k^{th}-order *label entropy* $H^\ell_k(t)$ of $t \in \mathcal{T}(\Sigma)$ is defined as

$$H^\ell_k(t) = \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} n^t_{z,a} \log\left(\frac{n^t_z}{n^t_{z,a}}\right),$$

where n^t_z and $n^t_{z,a}$ are from (7.1) and (7.2), respectively.

Intuitively, (normalized) k^{th}-order label entropy of a tree $t \in \mathcal{T}(\Sigma)$ tells us the expected uncertainty about the label of a node of t, given the labels of its k closest ancestors. We remark that in [32], it is not explicitly specified how to deal with nodes whose label history is shorter than k. The three natural variants (as in the case of the label-shape entropy, see Section 6.4) are

(i) padding the label histories with a symbol $\square \in \Sigma$ (this is our choice),

(ii) padding label histories with a symbol $\diamond \notin \Sigma$, or equivalently, allowing label histories of length smaller than k, and

(iii) ignoring nodes whose label history is shorter than k.

However, similar considerations as presented in [S6, Theorem 7] for the label-shape entropy show that these approaches yield the same k^{th}-order label entropy up to an additional additive term of at most $m^<(1 + 1/\ln(2) + \log(\sigma|t|/m^<))$, where $m^<$ is the number of nodes at depth less than k in t.

Moreover, we remark that in the original paper on label entropy [32], the authors quite often assume disjoint label alphabets for inner nodes and leaves, i.e., inner nodes are labeled with symbols from an alphabet Σ_1 while leaves are labeled with symbols from an alphabet Σ_2 with $\Sigma_1 \cap \Sigma_2 = \emptyset$. As it seems more natural not to assume disjoint alphabets for inner nodes and leaves, we will not make this assumption in the following and remark that several other papers on label entropy also do not make this assumption [46, 60].

Degree entropy. Another notion of empirical entropy for trees is the entropy measure from [60], which we call *degree entropy*. Degree entropy is primarily defined for unlabeled plane trees, as it ignores node labels. Nevertheless the definition works for trees $t \in \mathcal{T}(\Sigma)$ over any alphabet Σ.

Definition 7.2 (Degree entropy, [60])**.** For a tree $t \in \mathcal{T}(\Sigma)$, the degree entropy $H^{\deg}(t)$ is the 0^{th}-order entropy of the node degrees, where n_i^t is from (7.3):

$$H^{\deg}(t) = \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right).$$

Note that this definition does not take node labels into account. For the special case of unlabeled trees the following result was shown in [60]:

Theorem 7.3 ([60, Theorem 1])**.** *Let $t \in \mathcal{T}(\{a\})$ be an unlabeled plane tree. Then t can be represented in $H^{\deg}(t) + \mathcal{O}(|t| \log\log|t| / \log|t|)$ many bits.*

Label-degree entropy and degree-label entropy. Recently, two combinations of the label entropy from [32] and the degree entropy from [60] were proposed in [46]. We call these two entropy measures *label-degree entropy* and *degree-label entropy*. Both notions are defined for Σ-labeled plane trees.

Definition 7.4 (Label-degree entropy, [46])**.** Let $t \in \mathcal{T}(\Sigma)$. The k^{th}-order *label-degree entropy* $H_k^{\ell,\deg}(t)$ of t is defined as

$$H_k^{\ell,\deg}(t) = \sum_{z\in\Sigma^k} \sum_{a\in\Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log\left(\frac{n_{z,a}^t}{n_{z,i,a}^t}\right),$$

where $n_{z,a}^t$ and $n_{z,i,a}^t$ are from (7.2) and (7.5), respectively.

Definition 7.5 (Degree-label entropy, [46])**.** Let $t \in \mathcal{T}(\Sigma)$. The k^{th}-order *degree-label entropy* $H_k^{\deg,\ell}(t)$ of t is defined as

$$H_k^{\deg,\ell}(t) = \sum_{z\in\Sigma^k} \sum_{i=0}^{|t|} \sum_{a\in\Sigma} n_{z,i,a}^t \log\left(\frac{n_{z,i}^t}{n_{z,i,a}^t}\right),$$

where $n_{z,i}^t$ and $n_{z,i,a}^t$ are from (7.4) and (7.5), respectively.

(Normalized) label-degree entropy of order k of a tree $t \in \mathcal{T}(\Sigma)$ yields the expected uncertainty about the degree of a node of t, given its k-label history and its label. In the same way, (normalized) degree-label entropy of order k of a tree t yields the expected uncertainty about the label of a node, given its k-label history and its degree. In order to deal with nodes whose label history is shorter than k one can again choose one of the three alternatives (i)–(iii) mentioned above. In [46], variant (ii) is chosen, while the above definitions correspond to choice (i). However, similar considerations as presented in [S6, Theorem 7] show again that these approaches are basically equivalent, except for an additional additive term of at most $m^<(1/\ln(2) + \log(\sigma|t|/m^<))$ in the case of the degree-label entropy, respectively, $m^<(1/\ln(2) + \log|t|)$ in the case of the label-degree entropy, where $m^<$ is the number of nodes at depth less than k. In [46], the following lemma is shown, which relates the degree-label entropy to the label entropy from Definition 7.1 and the label-degree entropy to the degree entropy from Definition 7.2:

Lemma 7.6 ([46, Lemma 1])**.** *For every $t \in \mathcal{T}(\Sigma)$, we have $H_k^{\ell,\deg}(t) \le H^{\deg}(t)$ and $H_k^{\deg,\ell}(t) \le H_k^{\ell}(t)$.*

Moreover, one of the main results of [46] states the following bounds (an upper bound of the form $H^{\deg}(t) + H_k^{\ell}(t) + o(n\log\hat{\sigma})$ on the number of bits needed to represent a tree $t \in \mathcal{T}_n(\Sigma)$ is also shown in [60, Theorem 5]).

Theorem 7.7 ([46, Theorem 12])**.** *Let $t \in \mathcal{T}(\Sigma)$, with $\sigma \le |t|^{1-\alpha}$ for some $\alpha > 0$. Then t can be represented within the following bounds (in bits):*

$$H^{\deg}(t) + H_k^{\ell}(t) + \mathcal{O}\left(\frac{|t|k\log\hat{\sigma} + |t|\log\log_{\hat{\sigma}}|t|}{\log_{\hat{\sigma}}|t|}\right),$$
$$H_k^{\ell,\deg}(t) + H_k^{\ell}(t) + \mathcal{O}\left(\frac{|t|k\log\hat{\sigma} + |t|\log\log_{\hat{\sigma}}|t|}{\log_{\hat{\sigma}}|t|}\right),$$
$$H_k^{\deg,\ell}(t) + H^{\deg}(t) + \mathcal{O}\left(\frac{|t|k\log\hat{\sigma} + |t|\log\log_{\hat{\sigma}}|t|}{\log_{\hat{\sigma}}|t|}\right).$$

7.3 Theoretical comparison of the entropy bounds

As we have seen in Theorems 6.23 and Theorem 7.7 (see also Theorem 6.21), known entropy bounds for the number of bits needed to represent a Σ-labeled plane tree $t \in \mathcal{T}(\Sigma)$ are achievable in terms of

- $H_k^{\ell s}(t)$,
- $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$,
- $H^{\deg}(t) + H_k^{\deg,\ell}(t)$, and
- $H^{\deg}(t) + H_k^{\ell}(t)$,

where in all cases we have to add a lower-order term. The term $H^{\deg}(t)+H_k^{\ell}(t)$ is lower-bounded by $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$ and $H^{\deg}(t) + H_k^{\deg,\ell}(t)$ by Lemma 7.6. For the special case of unlabeled plane trees $t \in \mathcal{T}(\{a\})$, $H^{\deg}(t)$ (plus lower-order terms) is an upper bound on the encoding length (see Theorem 7.3). Thus, for the special case of unlabeled trees, we will also compare the entropy bounds to $H^{\deg}(t)$.

One of the main tools in order to obtain our results is the well-known log-sum inequality (recall our conventions $0 \cdot \log(0) = 0$ and $0 \cdot \log(x/0) = 0$ for $x \geq 0$).

Lemma 7.8 ([20], Theorem 2.7.1)**.** *Let $a_1, a_2, \ldots, a_j, b_1, b_2, \ldots, b_j \geq 0$ be non-negative real numbers. Moreover, let $a = \sum_{i=1}^{j} a_i$ and $b = \sum_{i=1}^{j} b_i$. Then*

$$a \log\left(\frac{b}{a}\right) \geq \sum_{i=1}^{j} a_i \log\left(\frac{b_i}{a_i}\right).$$

7.3.1 Unlabeled full binary trees

In this subsection, we consider unlabeled full binary trees, i.e., $t \in \mathcal{B}(\{a\})$ over the unary alphabet $\Sigma = \{a\}$. As $\Sigma = \{a\}$, the fixed dummy symbol used to pad k-label-shape histories and k-label histories is $\square = a$. We start with a simple lemma.

Lemma 7.9. *Let $t \in \mathcal{B}(\{a\})$ be an unlabeled full binary tree with n leaves (that is, $\|t\| = n$, respectively, $|t| = 2n-1$). Then $H^{\deg}(t) = H_k^{\ell,\deg}(t) = (2 - o(1))n$.*

Proof. Every full binary tree of size $2n-1$ consists of n nodes of degree 0 and $n-1$ nodes of degree 2 (independently of the shape of the full binary tree). Thus, we obtain

$$\begin{aligned} H^{\deg}(t) &= \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right) = n \log\Big(\frac{2n-1}{n}\Big) + (n-1)\log\Big(\frac{2n-1}{n-1}\Big) \\ &\geq 2n(1 - o(1)). \end{aligned}$$

Moreover, as t is unlabeled, every node has the same label and the same label history. Thus, $H_k^{\ell,\deg}(t) = H^{\deg}(t)$. □

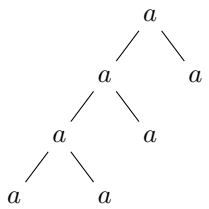


Figure 7.3: The tree t_4 from Lemma 7.10.

On the other hand, for the label-shape entropy we have:

Lemma 7.10. *There exists a sequence of unlabeled full binary trees* $(t_n)_{n\geq 1}$ *such that* $|t_n| = 2n-1$ *and* $H_k^{\ell s}(t_n) \leq \log(en)$ *for all* $n \geq 1$ *and* $1 \leq k \leq n$.

Proof. We define $t_1 = a$ and $t_n = a(t_{n-1}, a)$ for $n \geq 2$. Hence, t_n is a left-degenerate full binary tree with n leaves such that every node is labeled with the symbol a. Figure 7.3 shows t_4. The statement of the lemma follows by computing the label-shape entropy for t_n. □

Lemmas 7.9 and 7.10 already indicate that all empirical tree entropies considered in this chapter except for the label-shape entropy are not suitable entropy concepts for unlabeled full binary trees. For every unlabeled full binary tree $t \in \mathcal{B}(\{a\})$ with n leaves (and $2n-1$ nodes) we have:

- $H_k^{\ell}(t) = H_k^{\deg,\ell}(t) = 0$, as every node of t has the same label.
- $H_k^{\ell}(t) + H_k^{\ell,\deg}(t) = H^{\deg}(t) + H_k^{\deg,\ell}(t) = H_k^{\ell}(t) + H^{\deg}(t) = H^{\deg}(t)$ and these values are lower bounded by $2n(1-o(1))$ (Lemma 7.9).

The only notion of empirical tree entropy that is able to capture regularities in unlabeled full binary trees (and that attains different values for different full binary trees of the same size) is the label-shape entropy (Definition 6.14).

In fact, the definition of label-shape entropy is particularly motivated by the inability of the other entropy notions for measuring the compressibility of unlabeled full binary trees.

7.3.2 Labeled full binary trees

Next, we consider labeled full binary trees $t \in \mathcal{B}(\Sigma)$, where Σ is arbitrary. We start with a lemma that shows that the label-shape entropy lower-bounds all of the other entropy bounds in the case of labeled full binary trees.

Lemma 7.11. *Let* $t \in \mathcal{B}(\Sigma)$ *be a full binary tree. Then*

(i) $H_k^{\ell s}(t) \leq H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$ *and*

(ii) $H_k^{\ell s}(t) \leq H^{\deg}(t) + H_k^{\deg,\ell}(t)$.

Proof. We start with proving statement (i): We have

$$\begin{aligned}
H_k^{\ell s}(t) &= \sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{m^t_z}{m^t_{z,(a,i)}}\right)\\
&= \sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{m^t_z}{m^t_{z,(a,0)}+m^t_{z,(a,2)}}\cdot\frac{m^t_{z,(a,0)}+m^t_{z,(a,2)}}{m^t_{z,(a,i)}}\right)\\
&= \sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\left(m^t_{z,(a,0)}+m^t_{z,(a,2)}\right)\log\left(\frac{m^t_z}{m^t_{z,(a,0)}+m^t_{z,(a,2)}}\right)\\
&\quad+\sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{m^t_{z,(a,0)}+m^t_{z,(a,2)}}{m^t_{z,(a,i)}}\right)\\
&\le \sum_{z\in\Sigma^k}\sum_{a\in\Sigma} n^t_{z,a}\log\left(\frac{n^t_z}{n^t_{z,a}}\right)+\sum_{z\in\Sigma^k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} n^t_{z,i,a}\log\left(\frac{n^t_{z,a}}{n^t_{z,i,a}}\right)\\
&= H_k^{\ell}(t)+H_k^{\ell,\deg}(t),
\end{aligned}$$

where the inequality in the second last line follows from the log-sum inequality (Lemma 7.8) and the last equality follows from the fact that in a full binary tree, every node is either of degree 0 or 2. Statement (ii) can be shown similarly:

$$\begin{aligned}
H_k^{\ell s}(t) &= \sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{m^t_z}{m^t_{z,(a,i)}}\right)\\
&= \sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{m^t_z}{\sum_{a\in\Sigma} m^t_{z,(a,i)}}\cdot\frac{\sum_{a\in\Sigma} m^t_{z,(a,i)}}{m^t_{z,(a,i)}}\right)\\
&= \sum_{z\in\mathcal{H}_k}\sum_{i\in\{0,2\}}\left(\sum_{a\in\Sigma} m^t_{z,(a,i)}\right)\log\left(\frac{m^t_z}{\sum_{a\in\Sigma} m^t_{z,(a,i)}}\right)\\
&\quad+\sum_{z\in\mathcal{H}_k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} m^t_{z,(a,i)} \log\left(\frac{\sum_{a\in\Sigma} m^t_{z,(a,i)}}{m^t_{z,(a,i)}}\right)\\
&\le \sum_{i\in\{0,2\}} n^t_i\log\left(\frac{|t|}{n^t_i}\right)+\sum_{z\in\Sigma^k}\sum_{a\in\Sigma}\sum_{i\in\{0,2\}} n^t_{z,i,a}\log\left(\frac{n^t_{z,i}}{n^t_{z,i,a}}\right)\\
&= H^{\deg}(t)+H_k^{\deg,\ell}(t),
\end{aligned}$$

where the inequality follows again from the log-sum inequality. □

The inequality $H_k^{\ell s}(t) \le H_k^{\ell}(t) + H^{\deg}(t)$ for $t \in \mathcal{B}(\Sigma)$ now follows from Lemma 7.11 and Lemma 7.6. Moreover, by Lemma 7.9 and Lemma 7.10, we already know that there exist sequences $(t_n)_{n\ge1}$ of full binary trees, for which t_n has n leaves and $H_k^{\ell s}(t_n)$ is exponentially smaller than the other three entropy bounds. Moreover, $H^{\deg}(t) = 2n(1-o(1))$ for every $t \in \mathcal{B}_n(\Sigma)$, which implies $H_k^{\ell}(t) + H^{\deg}(t) \ge H^{\deg}(t) + H_k^{\deg,\ell}(t) \ge 2n(1-o(1))$.

7.3.3 Unlabeled plane trees

In this subsection, we consider plane trees $t \in \mathcal{T}(\Sigma)$ over the unary alphabet $\Sigma = \{a\}$. As $\Sigma = \{a\}$, the fixed dummy symbol used to pad k-label-shape histories and k-label histories is $\square = a$. Moreover, note that in order to compute $H_k^{\ell s}(t)$ for a plane tree $t \in \mathcal{T}(\Sigma)$, we have to consider $\mathrm{fcns}(t)$. Note that $\mathrm{fcns}(t)$ is then an unlabeled full binary tree: we take $\square = a$ by our conventions for the dummy symbol; hence the $\square$-labeled leaves in $\mathrm{fcns}(t)$ are labeled with a, too.

As in the case of unlabeled full binary trees, we observe that some entropy measures, in particular those that involve labels, only attain trivial values for plane unlabeled trees. More precisely, for every tree $t \in \mathcal{T}(\{a\})$ we have

- $H_k^{\ell}(t) = H_k^{\deg,\ell}(t) = 0$, as every node has the same label a, and
- $H^{\deg}(t) = H_k^{\ell,\deg}(t)$, as every node has the same k-label history and the same label.
- We get $H_k^{\ell}(t) + H_k^{\ell,\deg}(t) = H^{\deg}(t)$, $H^{\deg}(t) + H_k^{\deg,\ell}(t) = H^{\deg}(t)$ and $H^{\deg}(t) + H_k^{\ell}(t) = H^{\deg}(t)$.

By this observation, we only compare $H_k^{\ell s}(t)$ with $H^{\deg}(t)$ for $t \in \mathcal{T}(\{a\})$ in this subsection. By Lemma 7.9 and Lemma 7.10, there exists a sequence of unlabeled trees $(t_n)_{n\geq 1}$ such that $|t_n| = \Theta(n)$ and for which $H_k^{\ell s}(t_n)$ is exponentially smaller than $H^{\deg}(t_n)$.

In general, we find the following for unlabeled plane trees.

Theorem 7.12. *For every unlabeled plane tree $t \in \mathcal{T}(\{a\})$ with $|t| \geq 2$ and every integer $k \geq 1$, we have $H_k^{\ell s}(t) \leq 2H^{\deg}(t) + 2\log|t| + 4$.*

Proof. We start the proof with some simple counting facts with respect to the first-child next-sibling encoding. Consider a plane tree $t \in \mathcal{T}(\{a\})$ with $|t| \geq 2$. We claim that

(i) the number of inner nodes of $\mathrm{fcns}(t)$ which are left children equals the number of nodes of t of degree at least 1, and

(ii) the number of leaves of $\mathrm{fcns}(t)$ which are left children equals the number of nodes of t which are leaves.

To show this, recall that by definition of the first-child next-sibling encoding (Definition 2.13), every inner node of $\mathrm{fcns}(t)$ corresponds in a bijective way to a node of t. For an inner node v of $\mathrm{fcns}(t)$, we denote with $\mathrm{fcns}^{-1}(v)$ the corresponding node of t. In the same way, for a node v of t, we denote with $\mathrm{fcns}(v)$ the inner node of $\mathrm{fcns}(t)$ that corresponds to v.

The inner nodes of $\mathrm{fcns}(t)$ are moreover in bijective correspondence with the nodes of $\mathrm{fcns}(t)$ that are left children; the corresponding bijection is the function $\mathrm{parent}(\cdot)$ that maps a left child to its parent node. Hence, the composition of the mappings $\mathrm{parent}(\cdot)$ and $\mathrm{fcns}^{-1}(\cdot)$ can be viewed as a bijection from the left children in $\mathrm{fcns}(t)$ to the nodes of t.

Consider a left child v in $\mathrm{fcns}(t)$ and let $v' = \mathrm{fcns}^{-1}(\mathrm{parent}(v))$ be the corresponding node in t. If v is an inner node of $\mathrm{fcns}(t)$ then v' has a first child in t, i.e., its degree is at least one. On the other hand, if v is a leaf of $\mathrm{fcns}(t)$ then v' has no first child in t, i.e., its degree is zero. This yields the above statements (i) and (ii). Let us now fix $k \geq 1$ and let

$$\begin{aligned} \mathcal{H}_k^0 &= \{ai_1 \cdots ai_{k-1}a0 \mid i_1, \ldots, i_{k-1} \in \{0,1\}\} \subseteq \mathcal{H}_k \text{ and} \\ \mathcal{H}_k^1 &= \{ai_1 \cdots ai_{k-1}a1 \mid i_1, \ldots, i_{k-1} \in \{0,1\}\} \subseteq \mathcal{H}_k \end{aligned}$$

denote the sets of k-label-shape histories that end with 0, respectively 1. Let $n_{\geq 1}^t$ denote the number of nodes of t of degree at least 1 and for $z \in \mathcal{H}_k$ and $i \in \{0,2\}$ let $m_{z,i}^{\mathrm{fcns}(t)}$ denote the number of nodes in $\mathrm{fcns}(t)$ having k-label-shape history z and degree i. From (i) and (ii) we get

$$n_0^t = \sum_{z \in \mathcal{H}_k^0} m_{z,0}^{\mathrm{fcns}(t)} \quad \text{and} \quad n_{\geq 1}^t + 1 = \sum_{z \in \mathcal{H}_k^0} m_{z,2}^{\mathrm{fcns}(t)}. \tag{7.6}$$

The $+1$ in the second identity comes from the fact that on the right-hand side we also count the root node (which is not a left child of a node in $\mathrm{fcns}(t)$). Thus, we have

$$\begin{aligned} H^{\deg}(t) &= \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right) \geq n_{\geq 1}^t \log\left(\frac{|t|}{n_{\geq 1}^t}\right) + n_0^t \log\left(\frac{|t|}{n_0^t}\right) \\ &\geq (n_{\geq 1}^t + 1) \log\Big(\frac{|t|+1}{n_{\geq 1}^t + 1}\Big) - \log|t| + n_0^t \log\Big(\frac{|t|+1}{n_0^t}\Big) - \frac{n_0^t}{\ln(2)|t|}, \end{aligned}$$

where for the last inequality, we used $y/x \geq (y+1)/(x+1)$ if $y \geq x$ and

$$\log(|t|+1) - \log|t| \leq \frac{|t|+1-|t|}{\ln(2)|t|} = \frac{1}{\ln(2)|t|},$$

which follows from the mean value theorem. Hence, by the above equations (7.6) and the fact that $|t|+1$ equals the number of nodes v of $\mathrm{fcns}(t)$ with k-label-shape history $h_k^{\ell s}(v) \in \mathcal{H}_k^0$, we obtain

$$\begin{aligned} H^{\deg}(t) &\geq \Big(\sum_{z \in \mathcal{H}_k^0} m_{z,2}^{\mathrm{fcns}(t)}\Big) \log\left(\frac{\sum_{z \in \mathcal{H}_k^0} m_z^{\mathrm{fcns}(t)}}{\sum_{z \in \mathcal{H}_k^0} m_{z,2}^{\mathrm{fcns}(t)}}\right) \\ &\quad + \Big(\sum_{z \in \mathcal{H}_k^0} m_{z,0}^{\mathrm{fcns}(t)}\Big) \log\left(\frac{\sum_{z \in \mathcal{H}_k^0} m_z^{\mathrm{fcns}(t)}}{\sum_{z \in \mathcal{H}_k^0} m_{z,0}^{\mathrm{fcns}(t)}}\right) - \log|t| - 2 \\ &\geq \sum_{z \in \mathcal{H}_k^0} \sum_{i \in \{0,2\}} m_{z,i}^{\mathrm{fcns}(t)} \log\left(\frac{m_z^{\mathrm{fcns}(t)}}{m_{z,i}^{\mathrm{fcns}(t)}}\right) - \log|t| - 2, \end{aligned}$$

where the last inequality follows from the log-sum inequality (Lemma 7.8). In the next part of the proof, we establish a similar estimate by considering nodes

of fcns(t) with $h_k^{\ell s}(v) \in \mathcal{H}_k^1$. These nodes are exactly the right children in fcns(t) and there are $|t|$ such nodes. The composition of the parent-mapping and the mapping fcns^{-1} yields a bijection from the right children in fcns(t) to the nodes of t. Consider a node v in fcns(t) and assume that v is the right child of $v' = \text{parent}(v)$. If v is a leaf of fcns(t) then $\text{fcns}^{-1}(v')$ does not have a right sibling in t and if v is an inner node of fcns(t) then $\text{fcns}^{-1}(v')$ has a right sibling in t. Hence, the number of leaves v of fcns(t) with $h_k^{\ell s}(v) \in \mathcal{H}_k^1$ is equal to the number of nodes in t that do not have a right sibling. There are exactly $n_{\geq 1}^t + 1$ such nodes (there are $n_{\geq 1}^t$ nodes that are the right-most child of their parent node; in addition the root has no right sibling, too). Hence, we find

(iii) The number of leaves v of fcns(t) with $h_k^{\ell s}(v) \in \mathcal{H}_k^1$ equals one plus the number of nodes of t of degree at least 1:

$$\sum_{z \in \mathcal{H}_k^1} m_{z,0}^{\text{fcns}(t)} = n_{\geq 1}^t + 1.$$

(iv) For the number of leaves of $|t|$, we thus obtain:

$$n_0^t = |t| - \sum_{z \in \mathcal{H}_k^1} m_{z,0}^{\text{fcns}(t)} + 1 = \sum_{z \in \mathcal{H}_k^1} m_{z,2}^{\text{fcns}(t)} + 1.$$

Let $H(x) = x \log\left(\frac{|t|}{x}\right) + (|t| - x) \log\left(\frac{|t|}{|t|-x}\right)$ denote the binary entropy function (and recall that by convention, we have $0 \cdot \log(|t|/0) = 0$). Then the mapping $x \mapsto H(x) - H(x+1)$ with $(x \in [0, |t| - 1])$ is minimal for $x = 0$ and we find that $H(0) - H(1) = -\log|t| - (|t| - 1) \log\left(\frac{|t|}{|t|-1}\right) \geq -\log|t| - \log(e)$. We thus find

$$\begin{aligned} H^{\deg}(t) &= \sum_{i=0}^{n} n_i^t \log\left(\frac{|t|}{n_i^t}\right) \geq n_{\geq 1}^t \log\left(\frac{|t|}{n_{\geq 1}^t}\right) + n_0^t \log\left(\frac{|t|}{n_0^t}\right) \\ &\geq (n_{\geq 1}^t + 1) \log\left(\frac{|t|}{n_{\geq 1}^t + 1}\right) + (n_0^t - 1) \log\left(\frac{|t|}{n_0^t - 1}\right) - \log|t| - 2. \end{aligned}$$

By the above equations in (iii) and (iv), we thus get

$$\begin{aligned} H^{\deg}(t) &\geq \Big(\sum_{z \in \mathcal{H}_k^1} m_{z,0}^{\text{fcns}(t)} \Big) \log\left(\frac{\sum_{z \in \mathcal{H}_k^1} m_z^{\text{fcns}(t)}}{\sum_{z \in \mathcal{H}_k^1} m_{z,0}^{\text{fcns}(t)}} \right) \\ &\quad + \Big(\sum_{z \in \mathcal{H}_k^1} m_{z,2}^{\text{fcns}(t)} \Big) \log\left(\frac{\sum_{z \in \mathcal{H}_k^1} m_z^{\text{fcns}(t)}}{\sum_{z \in \mathcal{H}_k^1} m_{z,2}^{\text{fcns}(t)}} \right) - \log|t| - 2 \\ &\geq \sum_{z \in \mathcal{H}_k^1} \sum_{i \in \{0,2\}} m_{z,i}^{\text{fcns}(t)} \log\left(\frac{m_z^{\text{fcns}(t)}}{m_{z,i}^{\text{fcns}(t)}} \right) - \log|t| - 2, \end{aligned}$$

where the last inequality follows from the log-sum inequality. Altogether, since

$\mathcal{H}_k$ is the disjoint union of $\mathcal{H}_k^0$ and $\mathcal{H}_k^1$, we obtain:

$$H_k^{\ell s}(t) = \sum_{z \in \mathcal{H}_k} \sum_{i \in \{0,2\}} m_{z,i}^{\mathrm{fcns}(t)} \log \left(\frac{m_z^{\mathrm{fcns}(t)}}{m_{z,i}^{\mathrm{fcns}(t)}} \right) \leq 2H^{\deg}(t) + 2 \log |t| + 4.$$

This proves the theorem. □

It remains to remark that if we consider labeled trees over an alphabet Σ of size $\sigma > 1$, then there are families of trees, for which the degree entropy is asymptotically exponentially smaller than the k^{th}-order label-shape tree entropy. This is not very surprising, as the label-shape entropy incorporates the node labels, while the degree entropy does not.

7.3.4 Labeled plane trees

In this subsection, we consider Σ-labeled plane trees $t \in \mathcal{T}(\Sigma)$ over alphabets Σ of arbitrary size. The entropies to be compared in this general case are $H_k^{\ell s}(t)$, $H^{\deg}(t) + H_k^{\deg,\ell}(t)$, $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$ and $H^{\deg}(t) + H_k^{\ell}(t)$. Somewhat surprisingly it turns out that $H_k^{\ell}(t) + H_k^{\ell,\deg}(t)$ is at most $H^{\deg}(t) + H_k^{\deg,\ell}(t)$ for every tree t.

Theorem 7.13. *Let $t \in \mathcal{T}(\Sigma)$. Then $H_k^{\ell}(t) + H_k^{\ell,\deg}(t) \leq H^{\deg}(t) + H_k^{\deg,\ell}(t)$.*

Proof. We have

$$\begin{aligned}
& H_k^{\ell}(t) + H_k^{\ell,\deg}(t) \\
= & \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} n_{z,a}^t \log \left(\frac{n_z^t}{n_{z,a}^t} \right) + \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_{z,a}^t}{n_{z,i,a}^t} \right) \\
= & \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_z^t}{n_{z,a}^t} \right) + \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_{z,a}^t}{n_{z,i,a}^t} \right) \\
= & \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_z^t}{n_{z,i,a}^t} \right) \\
= & \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_z^t}{n_{z,i}^t} \right) + \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_{z,i}^t}{n_{z,i,a}^t} \right) \\
= & \sum_{z \in \Sigma^k} \sum_{i=0}^{|t|} n_{z,i}^t \log \left(\frac{n_z^t}{n_{z,i}^t} \right) + \sum_{z \in \Sigma^k} \sum_{a \in \Sigma} \sum_{i=0}^{|t|} n_{z,i,a}^t \log \left(\frac{n_{z,i}^t}{n_{z,i,a}^t} \right) \\
\leq & H^{\deg}(t) + H_k^{\deg,\ell}(t),
\end{aligned}$$

where the inequality follows from the log-sum inequality (Lemma 7.8). This proves the theorem. □

As a corollary of Theorem 7.13 it turns out that $H^{\deg}(t) + H_k^{\deg,\ell}(t)$ and $H_k^{\ell}(t) + H^{\deg}(t)$ are equivalent up to a constant factor.

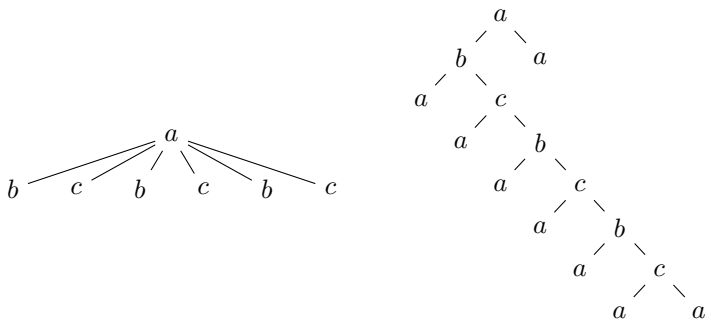


Figure 7.4: The labeled plane tree t_3 from Lemma 7.15 (left) and its first-child next-sibling encoding $\mathrm{fcns}(t_3)$ (right).

Corollary 7.14. *Let $t \in \mathcal{T}(\Sigma)$. Then*

$$H^{\deg}(t) + H_k^{\deg,\ell}(t) \leq H^{\deg}(t) + H_k^{\ell}(t) \leq 2H^{\deg}(t) + H_k^{\deg,\ell}(t).$$

Proof. The first inequality follows from Lemma 7.6. By Theorem 7.13, we have $H_k^{\ell}(t) \leq H^{\deg}(t) + H_k^{\deg,\ell}(t)$ from which the statement follows. □

In the rest of the section we present three examples showing that in all cases that are not covered by Theorem 7.13 we can achieve a non-constant (in most cases even exponential) separation between the corresponding entropy bounds.

Lemma 7.15. *There exists a sequence of labeled plane trees $(t_n)_{n\geq 1}$ such that for all $n \geq 1$ and $1 \leq k \leq 2n$:*

(i) $|t_n| = 2n + 1$,

(ii) $H_k^{\ell s}(t_n) \leq \log e + \log\left(n - \left\lfloor \frac{k-1}{2} \right\rfloor\right) + 2$,

(iii) $H_k^{\deg,\ell}(t_n) = 2n$ *and hence* $H^{\deg}(t_n) + H_k^{\deg,\ell}(t_n) \geq 2n$, *and*

(iv) $H_k^{\ell}(t_n) \geq 2n$ *and hence* $H_k^{\ell}(t_n) + H_k^{\ell,\deg}(t_n) \geq 2n$.

Proof. Define the labeled plane tree t_n as $t_n = a((bc)^n)$, that is, t_n is a tree consisting of a root node of degree $2n$ labeled with a and $2n$ leaves, of which n leaves are labeled b and n leaves are labeled c. The tree t_3 is depicted in Figure 7.4 on the left. Computing the entropy bounds for t_n proves the lemma (for more details, see [S5]). □

Lemma 7.15 shows that there are not only families of full binary trees, but also families of labeled plane (non-binary) trees $(t_n)_{n\geq 1}$ (for which we have to compute $H_k^{\ell s}(t_n)$ via the fcns-encoding) such that $|t_n| = \Theta(n)$ and $H_k^{\ell s}(t_n)$ is exponentially smaller than $H^{\deg}(t_n) + H_k^{\deg,\ell}(t_n)$ and $H_k^{\ell}(t_n) + H_k^{\ell,\deg}(t_n)$. The next lemma shows that there are also families of trees $(t_n)_{n\geq 1}$ such that $H_k^{\ell,\deg}(t_n) + H_k^{\ell}(t_n)$ is (even more than) exponentially smaller than $H^{\deg}(t_n) + H_k^{\deg,\ell}(t_n)$ (and thus, than $H^{\deg}(t_n) + H_k^{\ell}(t_n)$) and $H_k^{\ell s}(t_n)$).

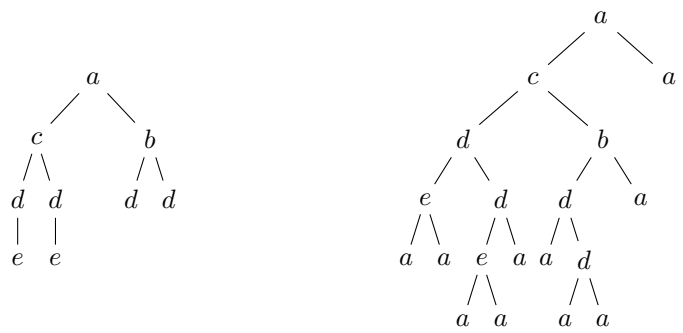


Figure 7.5: The labeled plane tree t_2 from Lemma 7.16 (left) and its first-child next-sibling encoding $\mathrm{fcns}(t_2)$ (right)

Lemma 7.16. *There exists a sequence of labeled plane trees $(t_n)_{n\geq 1}$ such that for all $n \geq 1$ and $1 \leq k \leq n$:*

(i) $|t_n| = 3n + 3$,

(ii) $H_k^{\ell s}(t_n) \geq 2(n - k + 1)$,

(iii) $H^{\deg}(t_n) + H_k^{\deg,\ell}(t_n) \geq 2n$ *and*

(iv) $H_1^{\ell}(t_n) + H_1^{\ell,\deg}(t_n) = 3\log(3)$.

Proof. Let $\Sigma = \{a, b, c, d, e\}$. We define the tree t_n as $t_n = a(c(d(e)^n)b(d^n))$. The tree t_2 is depicted in Figure 7.5 on the left. We have $|t_n| = 3n + 3$. Computing the entropy bounds for the tree t_n yields the statement of the lemma (for more details, see [S5]). □

Note that we clearly need $\Omega(\log n)$ bits to represent the tree t_n from the above proof (since we have to represent its size). This does not contradict Theorem 7.7 and the $\mathcal{O}(1)$-bound for $H_1^{\ell}(t_n) + H_1^{\ell,\deg}(t_n)$ in Lemma 7.16, since we have the additional additive term of order $o(|t|)$ in Theorem 7.7. In the following lemma, $n^{\underline{k}} = n(n-1)\cdots(n-k+1)$ denotes the falling factorial.

Lemma 7.17. *There exists a sequence of labeled plane trees $(t_{n,k})_{n\geq 1}$ (where $k(n) \leq n$ may depend on n) such that for all $n \geq 1$:*

(i) $|t_{n,k}| = 1 + n^{\underline{k}} + k \cdot n \cdot n^{\underline{k}}$,

(ii) $H^{\deg}(t_{n,k}) + H_1^{\ell}(t_{n,k}) \leq \mathcal{O}(n \cdot n^{\underline{k}} \cdot k \cdot \log k)$ *and*

(iii) $H_{k-1}^{\ell s}(t_{n,k}) \geq \Omega(n \cdot n^{\underline{k}} \cdot k \cdot \log(n - k + 1))$.

Proof. Let $[n]^{\underline{k}} = \{(i_1, i_2 \ldots, i_k) \mid 1 \leq i_1, \ldots, i_k \leq n, i_j \neq i_l \text{ for } j \neq l\}$. The tree $t_{n,k}$ is defined over the label alphabet

$$\Sigma_{n,k} = \{a\} \cup \{b_u \mid u \in [n]^{\underline{k}}\} \cup \{c_i \mid 1 \leq i \leq n\}$$

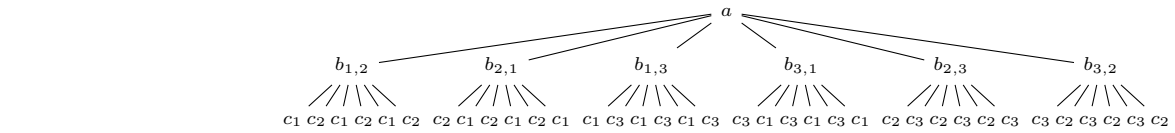


Figure 7.6: The tree $t_{3,2}$ from Lemma 7.17.

For $u = (i_1, i_2, \ldots, i_k) \in [n]^{\underline{k}}$ let us define $t_u = b_u((c_{i_1} c_{i_2} \cdots c_{i_k})^n)$ and let

$$t_{n,k} = a(t_{u_1} t_{u_2} \cdots t_{u_m}),$$

where $u_1, u_2, \ldots, u_m$ is an arbitrary enumeration of the set $[n]^{\underline{k}}$ (hence, $m = n^{\underline{k}}$). The tree $t_{3,2}$ is shown in Figure 7.6. Computing the entropy bounds for $t_{n,k}$ yields the statement of the lemma. The details of the computation of the entroy bounds are given in [S5]. □

If $k \in (\log n)^{\mathcal{O}(1)}$ then the trees $t_{n,k}$ from Lemma 7.17 satisfy

$$\frac{H^{\deg}(t_{n,k}) + H_1^{\ell}(t_{n,k})}{H_{k-1}^{\ell s}(t_{n,k})} \leq \mathcal{O}\left(\frac{\log k}{\log(n-k+1)}\right) = o(1). \tag{7.7}$$

7.4 Experimental comparison

In this section, we complement our theoretical results with experimental data. We computed the entropies $H^{\deg}$, $H_k^{\ell s}$, H_k^{ℓ}, $H_k^{\ell,\deg}$ and $H_k^{\deg,\ell}$ (for $k \in \{0, 1, 2, 4\}$) for 13 XML files from `http://xmlcompbench.sourceforge.net`. Table 7.1 shows the values for the entropy bounds $H_k^{\ell s}$, $H^{\deg} + H_k^{\ell}$, $H_k^{\ell} + H_k^{\ell,\deg}$ and $H^{\deg} + H_k^{\deg,\ell}$ (which can be achieved up to lower order terms by compressors, see [46] and Theorem 6.23). It turns out that for all XML trees considered in this comparison the k^{th}-order label-shape entropy (for $k > 0$) is significantly smaller than the entropy bounds from [46]. In Tables 7.2 and 7.3 we give the values for H_k^{ℓ}, $H_k^{\ell,\deg}$ and $H_k^{\deg,\ell}$ (divided by the tree size) for each XML file.

Additionally, we computed the label-shape entropy $H_k^{\ell s}$ for a modified version of each XML tree where all labels are replaced by a single dummy symbol, i.e., we considered the underlying, unlabeled tree as well (in Tables 7.2 and 7.3 this value is denoted by H_k^s, as only the shape of the tree is considered). Note again that the label-shape entropy $H_k^{\ell s}$ is the only measure for which this modification is interesting, because (i) the degree entropy $H^{\deg}$ is not affected since it does not take labels into account and (ii) we have $H_k^{\ell,\deg}(t) = H^{\deg}(t)$ and $H_k^{\ell}(t) = H_k^{\deg,\ell}(t) = 0$ for all unlabeled trees t and for all k. In the setting of unlabeled trees, our experimental data indicates that neither the label-shape entropy nor the degree entropy (which is the upper bound on the number of bits needed by the data structure in [60] ignoring lower order terms; see Theorem 7.3) is favorable.

XML	k	$H_k^{\ell s}$	$H^{\deg} + H_k^{\ell}$	$H_k^{\ell} + H_k^{\ell,\deg}$	$H^{\deg} + H_k^{\deg,\ell}$
BaseBall	0	202 568.08	153 814.94	146 066.64	146 066.64
	1	6 348.08	145 705.73	137 957.42	145 323.26
	2	2 671.95	145 705.73	137 957.42	145 323.26
	4	1 435.11	145 705.73	137 957.42	145 323.26
DBLP	0	18 727 523.44	14 576 781.00	12 967 501.16	12 967 501.16
	1	2 607 784.68	12 137 042.56	10 527 690.38	12 076 935.39
	2	2 076 410.50	12 136 974.71	10 527 595.96	12 076 845.69
	4	1 951 141.63	12 136 966.29	10 527 586.31	12 076 836.82
EXI-Array	0	1 098 274.54	962 858.05	649 410.59	649 410.59
	1	4 286.39	387 329.51	73 882.05	387 304.76
	2	4 270.18	387 329.51	73 882.05	387 304.76
	4	4 263.82	387 329.51	73 882.05	387 304.76
EXI-factbook	0	530 170.92	481 410.05	423 012.12	423 012.12
	1	11 772.65	239 499.01	181 101.08	204 649.84
	2	5 049.98	239 499.01	181 101.08	204 649.84
	4	4 345.42	239 499.01	181 101.08	204 649.84
EnWikiNew	0	2 118 359.59	1 877 639.22	1 384 034.65	1 384 034.65
	1	243 835.84	1 326 743.94	833 139.36	1 095 837.20
	2	78 689.86	1 326 743.94	833 139.36	1 095 837.20
	4	78 687.51	1 326 743.94	833 139.36	1 095 837.20
EnWikiQuote	0	1 372 201.38	1 229 530.04	894 768.55	894 768.55
	1	156 710.30	871 127.39	536 365.91	717 721.09
	2	51 557.50	871 127.39	536 365.91	717 721.09
	4	51 557.31	871 127.39	536 365.91	717 721.09
EnWikiVersity	0	2 568 158.43	2 264 856.93	1 644 997.36	1 644 997.36
	1	278 832.56	1 594 969.93	975 110.35	1 311 929.24
	2	74 456.55	1 594 969.93	975 110.35	1 311 929.24
	4	74 456.41	1 594 969.93	975 110.35	1 311 929.24
Nasa	0	3 022 100.11	2 872 172.41	2 214 641.55	2 214 641.55
	1	292 671.36	1 368 899.76	701 433.91	1 226 592.72
	2	168 551.10	1 363 699.16	696 194.53	1 221 474.16
	4	147 041.08	1 363 699.16	696 194.53	1 221 474.16
Shakespeare	0	655 517.90	521 889.47	395 890.85	395 890.85
	1	138 283.88	370 231.89	244 047.64	347 212.36
	2	125 837.77	370 061.20	243 843.87	347 041.31
	4	123 460.80	370 057.77	243 838.09	347 037.86
SwissProt	0	18 845 126.39	16 063 648.44	13 755 427.39	13 755 427.39
	1	3 051 570.48	11 065 924.67	8 757 703.61	10 238 734.83
	2	2 634 911.88	11 065 924.67	8 757 703.61	10 238 734.83
	4	2 314 609.48	11 065 924.67	8 757 703.61	10 238 734.83
Treebank	0	16 127 202.92	15 669 672.80	12 938 625.09	12 938 625.09
	1	7 504 481.18	12 301 414.61	9 482 695.67	9 925 567.44
	2	5 607 499.40	11 909 330.06	9 051 186.33	9 559 968.40
	4	4 675 093.61	11 626 935.89	8 736 301.14	9 285 544.85
USHouse	0	36 266.08	34 369.06	28 381.43	28 381.43
	1	10 490.44	24 249.78	17 968.41	19 438.19
	2	9 079.97	24 037.34	17 569.59	19 216.99
	4	6 308.98	23 634.87	16 830.00	18 783.36
XMark1	0	1 250 525.41	1 186 214.34	988 678.93	988 678.93
	1	167 586.81	592 634.17	394 639.43	523 996.29
	2	131 057.35	592 625.76	394 565.79	523 969.97
	4	127 157.34	592 037.39	393 770.73	523 432.87

Table 7.1: A comparison of the upper bounds on the number of bits used by the tree representation from Theorem 6.23 (third column) and the data structure from [46] (columns 4, 5 and 6), where lower order terms are ignored.

XML	n	$H^{\deg}/n$	k	H_k^s/n	$H_k^{\ell s}/n$	H_k^{ℓ}/n	$H_k^{\deg,\ell}/n$	$H_k^{\ell,\deg}/n$
BaseBall	28 306	0.2777	0	2.0000	7.1564	5.1563	4.8826	0.0039
			1	0.5271	0.2243	4.8698	4.8563	0.0039
			2	0.5218	0.0944	4.8698	4.8563	0.0039
			4	0.5122	0.0507	4.8698	4.8563	0.0039
DBLP	3 332 130	0.7543	0	2.0000	5.6203	3.6203	3.1373	0.2714
			1	0.9343	0.7826	2.8881	2.8701	0.2713
			2	0.9064	0.6231	2.8881	2.8700	0.2713
			4	0.8340	0.5856	2.8881	2.8700	0.2713
EXI-Array	226 523	1.4022	0	2.0000	4.8484	2.8484	1.4647	0.0185
			1	1.9736	0.0189	0.3077	0.3076	0.0185
			2	1.8227	0.0189	0.3077	0.3076	0.0185
			4	0.4486	0.0188	0.3077	0.3076	0.0185
EXI-factbook	55 453	1.1207	0	2.0000	9.5607	7.5607	6.5076	0.0676
			1	1.2641	0.2123	3.1983	2.5698	0.0676
			2	1.2319	0.0911	3.1983	2.5698	0.0676
			4	1.1811	0.0784	3.1983	2.5698	0.0676
EnWikiNew	404 652	1.4051	0	2.0000	5.2350	3.2350	2.0152	0.1853
			1	1.6514	0.6026	1.8736	1.3030	0.1853
			2	1.3977	0.1945	1.8736	1.3030	0.1853
			4	1.0771	0.1945	1.8736	1.3030	0.1853
EnWikiQuote	262 955	1.4574	0	2.0000	5.2184	3.2184	1.9453	0.1844
			1	1.6878	0.5960	1.8554	1.2720	0.1844
			2	1.4695	0.1961	1.8554	1.2720	0.1844
			4	1.0229	0.1961	1.8554	1.2720	0.1844
EnWikiVersity	495 839	1.3883	0	2.0000	5.1794	3.1794	1.9293	0.1382
			1	1.6647	0.5623	1.8284	1.2576	0.1382
			2	1.4106	0.1502	1.8284	1.2576	0.1382
			4	0.9645	0.1502	1.8284	1.2576	0.1382

Table 7.2: Experimental results for XML tree structures, where n is the number of nodes and H_k^s is the label-shape entropy for the underlying, unlabeled tree.

XML	n	$H^{\deg}/n$	k	H_k^s/n	$H_k^{\ell s}/n$	H_k^{ℓ}/n	$H_k^{\deg,\ell}/n$	$H_k^{\ell,\deg}/n$
Nasa	476 646	1.6855	0	2.0000	6.3403	4.3403	2.9608	0.3060
			1	1.8834	0.6140	1.1865	0.8879	0.2851
			2	1.8483	0.3536	1.1756	0.8772	0.2850
			4	1.3824	0.3085	1.1756	0.8772	0.2850
Shakespeare	179 690	1.2563	0	2.0000	3.6480	1.6480	0.9468	0.5551
			1	1.3713	0.7696	0.8040	0.6759	0.5541
			2	1.2713	0.7003	0.8031	0.6750	0.5539
			4	1.1215	0.6871	0.8031	0.6750	0.5539
SwissProt	2 977 031	1.0657	0	2.0000	6.3302	4.3302	3.5548	0.2903
			1	1.2108	1.0250	2.6514	2.3736	0.2903
			2	1.0730	0.8851	2.6514	2.3736	0.2903
			4	1.0553	0.7775	2.6514	2.3736	0.2903
Treebank	2 437 666	1.8123	0	2.0000	6.6158	4.6158	3.4955	0.6920
			1	1.9707	3.0786	3.2341	2.2594	0.6560
			2	1.8014	2.3004	3.0732	2.1095	0.6398
			4	1.7620	1.9179	2.9574	1.9969	0.6265
USHouse	6 712	1.7175	0	2.0000	5.4032	3.4030	2.5109	0.8254
			1	1.8263	1.5629	1.8954	1.1785	0.7817
			2	1.5810	1.3528	1.8637	1.1456	0.7539
			4	1.2958	0.9400	1.8038	1.0810	0.7037
XMark1	167 865	1.6169	0	2.0000	7.4496	5.4496	4.2728	0.4401
			1	1.6917	0.9983	1.9135	1.5046	0.4374
			2	1.6820	0.7807	1.9135	1.5045	0.4370
			4	1.5735	0.7575	1.9100	1.5013	0.4358

Table 7.3: Experimental results for the second part of the XML tree structures, where n denotes the number of nodes and H_k^s is the label-shape entropy for the underlying, unlabeled tree.

7.5 Conclusion and open problems

We have carried out a systematic comparison of several existing notions of empirical tree entropy and underpinned our theoretical investigations by experimental results.

The separation between $H_k^{\ell s}$ and $H_1^{\ell} + H^{\deg}$ achieved in Lemma 7.17 is quite weak: for a constant k, $H_k^{\ell s}$ is only by a logarithmic factor larger than $H_1^{\ell} + H^{\deg}$; see (7.7). In contrast, in Lemma 7.15 and Lemma 7.16 we achieved an exponential separation. It remains open, whether such an exponential separation is also possible for $H_k^{\ell s}$ and $H_1^{\ell} + H^{\deg}$. In other words, does there exist a sequence of trees $(t_n)_{n\geq 1}$ such that $H_k^{\ell s}(t_n) \geq \Omega(n)$ and $H_1^{\ell}(t_n) + H^{\deg}(t_n) \leq \mathcal{O}(\log n)$?

Let us remark that Ganczorz's succinct tree representations [46] that achieve (up to lower-order terms) the entropy bounds $H_k^{\ell} + H_k^{\ell,\deg}$ and $H_{\deg} + H_k^{\deg,\ell}$ allow *constant* query times for a large number of tree queries. For the label-shape entropy $H_k^{\ell s}$, such a result is not known for labeled trees (over an alphabet of arbitrary size). Our tree representation based on tree straight-line programs from Chapter 6 that achieves an entropy bound in terms of the label-shape entropy can be queried in logarithmic time (under the assumption that the height of the grammar is logarithmic, which can be enforced by [43]). For unlabeled plane trees, we present a tree representation that achieves an entropy bound in terms of the label-shape entropy and which can be queried in constant time in Chapter 9.

Finally, it remains to remark that in general, various other notions of empirical tree entropy that have not been introduced and considered in the literature yet are possible. Future research might be to investigate and compare these various possibilities for entropy notions of empirical tree entropy, in particular, with respect to how well they capture regularities and compressibility of real-world tree data sets.

Part III

Hypersuccinct Trees

Chapter 8

Hypersuccinct binary trees

8.1 Introduction

In this chapter, we focus on universal source coding aspects of tree compression. So far, we have covered two tree compression formalisms in this work, compression by minimal DAGs (Chapters 3 to 5) and its generalization, grammar-based tree compression by TSLPs (Chapter 6). For the tree encoder based on TSLPs from Chapter 6, we have already obtained a universality result in Section 6.5. Our technical main result of Chapter 6, Theorem 6.20, states that for all labeled full binary trees $t \in \mathcal{B}(\Sigma)$ and k^{th}-order label-shape processes $\mathcal{P}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$, the encoding length $|E_\Psi(t)|$ of the tree encoder E_Ψ (consisting of a suitable grammar-based compressor $t \to \mathcal{G}_t$ together with the binary encoding B of the grammar from Section 6.3) is upper-bounded in terms of the *self-information* $\log(1/\mathsf{Prob}_{\mathcal{P}}(t))$ of t plus lower-order terms (if k is small enough). In other words, E_Ψ is *worst-case universal* with respect to the class of all k^{th}-order label-shape processes.

In previous works [105], [S3], universal tree source coding has been considered with respect to the family of unlabeled full binary trees, and mainly with respect to two types of tree sources, which are leaf-centric binary tree sources (as defined in Definition 3.1 in Chapter 3) and depth-centric binary tree sources, which closely resemble leaf-centric binary tree sources, except that they induce a probability distribution on the set of full binary trees of fixed depth (a formal definition follows in Section 8.4).

In [105], a compressed tree representation based on DAG-compression is considered. For this, the minimal DAG of the input tree is encoded by a binary string; this encoding step is similar to the binary coding of SLPs from [64]. This yields a tree encoder $E_{\mathrm{dag}} \colon \mathcal{B} \to \{0,1\}^*$. In order to show universality results for the tree encoder E_{dag}, the *maximal pointwise redundancy* (also called *worst-case redundancy*) and the *average-case redundancy* of E_{dag} are investigated in [105] and [S3]. The worst-case redundancy (respectively, average-case redundancy) of an encoding with respect to a given source measures the maximal (respectively, average) additive deviation of the code length from the self information with

respect to the source, normalized by the size of the encoded objects. That is, the worst-case redundancy of E_{dag} with respect to the leaf-centric or depth-centric binary tree source ℓ is defined as

$$\max_{t\in\mathcal{S}_n, P_\ell(t)>0} \frac{1}{\|t\|} \left(|E_{\mathrm{dag}}(t)| + \log(P_\ell(t))\right), \tag{8.1}$$

where $P_\ell(t)$ denotes the probability assigned to a tree $t \in \mathcal{B}$ by the source ℓ, and $\mathcal{S}_n$ is the set of full binary trees of leafsize n, respectively depth n, if ℓ is a leaf-centric, respectively, depth-centric binary tree source. Similarly, the average-case redundancy of E_{dag} with respect to the leaf-centric or depth-centric binary tree source ℓ is defined as

$$\sum_{t\in\mathcal{S}_n, P_\ell(t)>0} \frac{P_\ell(t)}{\|t\|} \left(|E_{\mathrm{dag}}(t)| + \log(P_\ell(t))\right). \tag{8.2}$$

Let $|\mathfrak{D}(t)|$ denote the size of the minimal DAG of a tree t. Specifically, it is shown in [105], [S3] that for all for all leaf-centric (respectively, depth-centric) binary tree sources ℓ, for which the average compression ratio achieved by the minimal DAG (i.e., the sum $\sum P_\ell(t)|\mathfrak{D}(t)|/\|t\|$, where the summation ranges over all $t \in \mathcal{B}_n$, respectively, all full binary trees t of depth n) converges to zero for $n \to \infty$, the average-case redundancy of E_{dag} converges to zero as $n \to \infty$. Note that this yields an immediate application of the results shown in Chapter 3 of this work, where we have presented several classes of leaf-centric binary tree sources that satisfy this property.

Moreover, for all leaf-centric (respectively, depth-centric) binary tree sources ℓ, for which the worst-case compression ratio achieved by the minimal DAG (i.e., the maximum of $|\mathfrak{D}(t)|/\|t\|$ taken over all trees $t \in \mathcal{B}_n$, respectively, over all full binary trees t of depth n) converges to zero as $n \to \infty$, the worst-case redundancy of E_{dag} converges to zero as $n \to \infty$.

Furthermore, in [S3], universal tree source coding based on TSLPs is considered. For this, the binary encoding of SLPs and DAGs from [64, 105] is extended to TSLPs, which yields a tree encoder $E_{\mathrm{tslp}} \colon \mathcal{B} \to \{0,1\}^*$. In fact, the tree encoder E_{tslp} investigated in [S3] is almost the same as the tree encoder E_Ψ from Chapter 6 of this work, except for the fact that E_{tslp} only deals with unlabeled full binary trees. The tree encoder E_{tslp} is then shown to be universal with respect to a particular class of leaf-centric, respectively, depth-centric binary tree sources, which we call *monotonic* in this work (see Definition 8.22), in the sense that its worst-case redundancy converges to zero. Specifically, the classes of leaf-centric, respectively, depth-centric tree sources, with respect to which the tree encoder E_{dag} and E_{tslp} are shown to be universal, are orthogonal [S3].

In this chapter, we present a new compressed tree encoding for unlabeled binary trees $t \in \mathcal{B}^\diamond$, which can be shown to be universal with respect to a large number of tree sources, and which in particular achieves optimal compression to within lower order terms for most binary tree sources covered by existing universal codes. We call this compressed tree representation *hypersuccinct trees*

(as an escalation of the name "ultrasuccinct trees", which is used for the tree representation presented in [60]). It is based on the tree decomposition technique introduced by Farzan and Munro in [30].

In particular, hypersuccinct trees are (worst-case or average-case) universal with respect to almost all classes of leaf-centric and depth-centric binary tree sources, for which the tree encoder E_{dag} and E_{tslp} from [105], [S3] are shown to be universal. Additionally, universality results can be shown with respect to two further types of tree sources, which are *node-type sources* (defined in Section 8.3) and *tame uniform-subclass sources* (which are covered in [S7]). From these universality results it follows that hypersuccinct trees simultaneously achieve the optimal (worst-case or average-case) space usage to within lower order terms for a wide range of distributions over tree shapes, including random binary search trees (Definition 2.15), random fringe-balanced binary search trees [103], binary trees with a given number of binary/unary/leaf nodes, (uniformly random) full binary trees, unary paths, uniformly chosen weight-balanced binary search trees (in the sense of BB[α]-trees, [88]), AVL trees and left-leaning red-black trees.

What is more, in contrast to the universal tree encodings considered in [105] and [S3], the hypersuccinct tree encoding can be turned into a compressed data structure that supports answering many navigational queries on the compressed representation in *constant* time on the word-RAM; recent lower bounds [95] imply that constant query times are not achievable for tree data structures based on grammar-based tree compression. Compared to prior work on succinct data structures [30, 60], we do not have to tailor our data structure to specific applications; from our universality results it follows that hypersuccinct trees automatically adapt to the trees at hand.

Specifically, with the hypersuccinct tree data structure, we solve an open problem for succinct range-minimum queries (RMQ). Here the task is to build a data structure from an array $A[1..n]$ of comparable items at preprocessing time that can answer subsequent queries without inspecting A again. The answer to the query $\mathrm{RMQ}(i, j)$, for $1 \leq i \leq j \leq n$, is the index (in A) of the (leftmost) minimum in $A[i..j]$. The hypersuccinct tree data structure answers RMQ in constant time using the optimal expected space of $1.736n + o(n)$ bits when the array is a random permutation (and $2n + o(n)$ in the worst case); previous work either had suboptimal space [22] or $\Omega(n)$ query time [50].

In this work, we focus on the universality aspects of hypersuccinct trees. For data structure aspects of hypersuccinct trees and a detailed comparison of hypersuccinct trees with the state of the art in the field of tree data structures, we refer to [S7]. Tree data structures have been extensively studied in the literature, see, e.g. [84, 98] for an overview. In Section 8.2, we introduce our hypersuccinct tree encoding for binary trees. In Section 8.3 and Section 8.4, we present a selection of the universality results that hold for hypersuccinct trees. Further universality results, which are closer related to data structure aspects of tree compression (as the optimal compression of AVL-trees and red-black trees), are shown in [S7].

The results of this chapter are published in [S7].

8.2 Hypersuccinct encoding of binary trees

In this chapter, we focus on the family of unlabeled binary trees $\mathcal{B}^\diamond$ (as defined in Definition 2.3). In particular, this includes the family of unlabeled full binary trees $\mathcal{B}$ (defined in Definition 2.2) as a special case, respectively, most concepts considered in this chapter can be naturally transferred to the family of full binary trees via the natural one-to-one correspondence between the sets $\mathcal{B}$ and $\mathcal{B}^\diamond$.

Sometimes it will be convenient to consider the *empty tree*, which is a tree of size zero. Let $t \in \mathcal{B}^\diamond$ and let v be a node of t. Recall that $t_l[v]$ (respectively, $t_r[v]$) denotes the fringe subtree rooted in v's left (respectively, right) child. If v does not have a left (respectively, right) child, then $t_l[v]$ (respectively, $t_r[v]$) is the empty tree. If v is the root of t, we again write t_l and t_r for $t_l[v]$ and $t_r[v]$.

A binary tree $t \in \mathcal{B}^\diamond$ of size n can be encoded using $2n$ bits via the balanced parenthesis encoding:

Definition 8.1 (Balanced parenthesis encoding)**.** We define the *balanced parenthesis encoding* for binary trees $BP\colon \mathcal{B}^\diamond \to \{(,)\}^*$ inductively by

$$BP(t) = \begin{cases} \epsilon & \text{if } t \text{ is the empty tree,} \\ (\,BP(t_l)\,)\,BP(t_r) & \text{otherwise.} \end{cases}$$

Here, ϵ again denotes the empty string.

In this chapter, we make use of some well-known concepts of information theory, such as *Huffman coding* [20, 55], *arithmetic coding* [20, 104] and the *Elias gamma code* [29], which encodes a number $n \in \mathbb{N}$ as a binary string using $2\lfloor \log n \rfloor + 1$ bits. With $\gamma(n)$ we denote the Elias gamma encoding of $n \in \mathbb{N}$.

Tree decomposition algorithm. Our compressed tree code is based on the Farzan-Munro algorithm [30] to decompose a tree into connected subtrees, which we call *micro trees*. This algorithm was originally designed for plane trees; we state its properties here when applied on binary trees. The results follow directly from the result proven in [30] and the fact that node degrees are at most two.

Lemma 8.2 (Binary tree decomposition, [30, Theorem 1])**.** *For any parameter $\kappa \geq 1$, a binary tree $t \in \mathcal{B}^\diamond$ with n nodes can be decomposed, in linear time, into $\Theta(n/\kappa)$ pairwise disjoint subtrees (called micro trees) of $\leq 2\kappa$ nodes each. Each of these micro trees has at most three connections to other micro trees:*

- *an edge from a parent micro tree to the root of the micro tree,*
- *an edge to another micro tree in the left subtree of the micro tree root,*
- *an edge to another micro tree in the right subtree of the micro tree root.*
- *At least one of the edges to a child micro tree (if both of them exist) emanates from the root itself.*

In particular, contracting micro trees into single nodes yields again a binary tree.

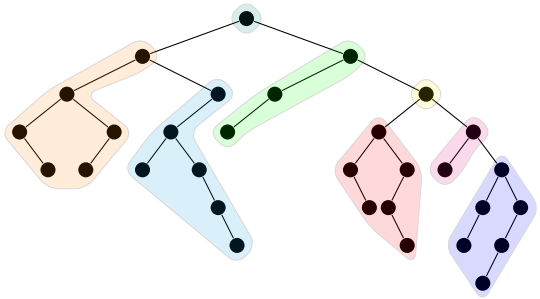


Figure 8.1: A binary tree with $n = 31$ nodes, decomposed into micro trees by the tree decomposition algorithm from [30] with $\kappa = 6$.

A binary tree decomposed into micro trees according to the decomposition algorithm from [30] can be found in Figure 8.1. If a node v's parent u belongs to a different micro tree, u will have a "null pointer" within its micro tree, that is, it loses its child there. To recover these connections between micro trees, we do not only need the information which micro tree is a child of which other micro tree, but also which null pointer inside a micro tree leads to the lost child. We refer to this null pointer as the *portal* of the (parent) micro tree (to the child micro tree). An additional property that we need is stated in the following lemma; it follows directly from the construction of micro trees in the tree decomposition algorithm [30].

Lemma 8.3. *Let v be the root of a micro tree constructed using the tree parameter κ, respectively, any ancestor of a micro tree root. Then $|t[v]| \geq \kappa$.*

Hypersuccinct code for binary trees. Based on the above properties of the tree partitioning algorithm from [30], we design an encoding $\mathsf{H}\colon \mathcal{B}^{\diamond} \to \{0,1\}^*$ for binary trees. Given a binary tree t of size n, we apply the Farzan-Munro algorithm with parameter $\kappa = \lceil \frac{1}{8}\log(n) \rceil$ to decompose the tree into micro trees $\mathsf{t}_1, \ldots, \mathsf{t}_m$, where $m = \Theta(n/\log n)$. The size of the micro trees $\mathsf{t}_1, \ldots, \mathsf{t}_m$ is thus upper-bounded by $s_{\max} = \lceil \frac{1}{4}\log(n) \rceil$. With Υ we denote the *top tier* of the tree t, which is obtained from t by contracting each micro tree t_i into a single node. In particular, as each micro tree t_i has at most 3 connections to other micro trees (a parent micro tree and (up to) two child micro trees, see Lemma 8.2), Υ is again a binary tree, and the size of Υ equals the number m of micro trees. With $\Sigma_{s_{\max}} \subseteq \bigcup_{k \leq s_{\max}} \mathcal{B}^{\diamond}_k$ we denote the set of shapes of micro trees that occur in the tree t. We observe that because of the limited size of micro trees, there are fewer different possible shapes of binary trees than we have micro trees.

The crucial idea of the hypersuccinct encoding is to treat each shape of a micro tree as a letter in the alphabet $\Sigma_{s_{\max}}$ and to compute a Huffman code $\Psi\colon \Sigma_{s_{\max}} \to \{0,1\}^*$ based on the frequency of occurrences of micro tree shapes in the sequence $(\mathsf{t}_1, \ldots, \mathsf{t}_m) \in \Sigma^m_{s_{\max}}$. For our hypersuccinct code, we then use a *length-restricted* version $\bar{\Psi}\colon \Sigma_{s_{\max}} \to \{0,1\}^*$ obtained from Ψ using a simple cutoff technique:

Definition 8.4 (Worst-case bounding trick)**.** Let $\Psi\colon \mathcal{B}^\diamond \to \{0,1\}^*$ denote a uniquely decodable encoding of binary trees. We define a simple *length-restricted version* $\bar{\Psi}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ of the binary-tree code Ψ as follows:

$$\bar{\Psi}(t) = \begin{cases} 0 \cdot \gamma(|t|+1) \cdot BP(t), & \text{if } |\Psi(t)| > 2|t| + 2\lfloor \log(|t|+1) \rfloor; \\ 1 \cdot \Psi(t), & \text{otherwise.} \end{cases}$$

Here, $BP(t)$ again denotes the balanced parenthesis encoding of t (as defined in Definition 8.1). The length-restricted code $\bar{\Psi}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ then uses

$$|\bar{\Psi}(t)| \leq \min\{|\Psi(t)|, 2|t| + 2\lfloor \log(|t|+1) \rfloor + 1\} + 1 \tag{8.3}$$

many bits in order to encode a binary tree $t \in \mathcal{B}^\diamond$ of size $|t|$.

Finally, for each micro tree, we have to encode which null pointers are portals to left and right child components (if they exist). For that, we store the portals' rank in the micro-tree-local left-to-right order of the null pointers using $\lceil \log(s_{\max}+1) \rceil$ bits each. We can thus encode t as follows:

(i) Store n and m in Elias gamma code,

(ii) followed by the balanced parenthesis bitstring for Υ.

(iii) Next comes an encoding for the length-restricted Huffman code $\bar{\Psi}$; for simplicity, we simply list all possible codewords and their corresponding binary trees by storing the size (in Elias gamma code) followed by their balanced parenthesis sequence.

(iv) Then, we list the length-restricted Huffman codes $\bar{\Psi}(\mathsf{t}_i)$ of all micro trees in depth-first order (of Υ).

(v) Finally, we store 2 $\lceil \log(s_{\max}+1) \rceil$-bit integers to encode the portal nulls for each micro tree, again in depth-first order (of Υ).

Altogether, this yields our *hypersuccinct encoding* $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ for binary trees. Decoding is obviously possible by first recovering the integers n, m, and the top tier Υ from its balanced parenthesis string, then reading the Huffman code and finally replacing each node in Υ by its micro tree in a depth-first traversal, using the information about portals to identify nodes from components that are adjacent in Υ. With respect to the length of the hypersuccinct code, we obtain the following lemma.

Lemma 8.5. *Let $t \in \mathcal{B}_n^\diamond$ be a binary tree of n nodes, decomposed into micro trees $\mathsf{t}_1, \ldots, \mathsf{t}_m$ by the Farzan-Munro algorithm. Let Ψ be an ordinary Huffman code for the string $\mathsf{t}_1 \ldots \mathsf{t}_m$. Then the hypersuccinct code encodes t with a binary codeword of length*

$$|\mathsf{H}(t)| \leq \sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right).$$

$\texttt{parent}(v)$	the parent of v, same as $\texttt{ancestor}(v, 1)$
$\texttt{deg}(v)$	the number of children of v
$\texttt{leftchild}(v)$	the left child of node v
$\texttt{rightchild}(v)$	the right child of node v
$\texttt{depth}(v)$	the depth of v (the number of edges between the root and v)
$\texttt{ancestor}(v, i)$	the ancestor of node v at depth $\texttt{depth}(v) - i$
$\texttt{descendants}(v)$	the number of descendants of v
$\texttt{height}(v)$	the height of the subtree rooted at node v
$\texttt{LCA}(v, u)$	the lowest common ancestor of nodes u and v
$\texttt{leftmostleaf}(v)$	the leftmost leaf descendant of v
$\texttt{rightmostleaf}(v)$	the rightmost leaf descendant of v
$\texttt{levelleftmost}(i)$	the leftmost node on level i
$\texttt{levelrightmost}(i)$	the rightmost node on level i
$\texttt{levelpred}(v)$	the node immediately to the left of v on the same level
$\texttt{levelsucc}(v)$	the node immediately to the right of v on the same level
$\texttt{noderank}_X(v)$	the position of v in the X-order, $X \in \{\texttt{pre}, \texttt{post}, \texttt{in}\}$
$\texttt{select}_X(i)$	the ith node in the X-order, $X \in \{\texttt{pre}, \texttt{post}, \texttt{in}\}$
$\texttt{leafrank}(v)$	the number of leaves before and including v in preorder
$\texttt{leafselect}(i)$	the ith leaf in preorder

Table 8.1: Navigational operations on succinct binary trees (v denotes a node and i an integer).

Proof. We first show that, among the five parts of the hypersuccinct binary-tree code for $t \in \mathcal{B}_n^\diamond$, all but the second to last one contribute $\mathcal{O}(n \log\log n / \log n)$ bits. Part (i) clearly needs $\mathcal{O}(\log n)$ bits and Part (ii) requires $2m = \Theta(n/\log n)$ bits. For Part (iii), observe that

$$|\Sigma_{s_{\max}}| \leq \sum_{k \leq \lceil \log n/4 \rceil} 4^k < \frac{4}{3} \cdot 4^{\log n/4+1} = \frac{16}{3}\sqrt{n}.$$

Together with the worst-case cutoff technique from Definition 8.4, we find that $|\bar{\Psi}(\mathfrak{t}_i)| \leq 2+2\log(s_{\max}+1)+2s_{\max} \leq \mathcal{O}(s_{\max})$, so we need asymptotically $\mathcal{O}(\sqrt{n})$ entries / codewords in the table, each of size $\mathcal{O}(s_{\max}) = \mathcal{O}(\log n)$, for an overall table size of $\mathcal{O}(\sqrt{n}\log n)$. Part (v) uses $m \cdot 2\lceil \log(s_{\max}+1) \rceil = \Theta(n \log\log n/\log n)$ bits of space. It remains to analyze Part (iv). We note that by applying the worst-case pruning scheme of Definition 8.4, we waste 1 bit per micro tree compared to a pure, non-restricted Huffman code. But these wasted bits amount to $m = \mathcal{O}(n/\log n)$ bits in total, and so are again a lower-order term:

$$\begin{aligned}\sum_{i=1}^{m} |\bar{\Psi}(\mathfrak{t}_i)| &= \sum_{i=1}^{m} \min\{|\Psi(\mathfrak{t}_i)| + 1, 2|\mathfrak{t}_i| + 2\lfloor \log|\mathfrak{t}_i| + 1 \rfloor + 2\} \\ &\leq \sum_{i=1}^{m} (|\Psi(\mathfrak{t}_i)| + 1) = \sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| + \mathcal{O}(n/\log n),\end{aligned}$$

where the first equality comes from Definition 8.4. This finishes the proof. □

What sets hypersuccinct code apart from other known universal codes is that it can be turned into a compressed tree data structure with constant-time queries. For the data structure details of hypersuccinct trees, we refer to [S7]. Specifically, Table 8.1 shows which queries can be supported by hypersuccinct trees in constant time on the word-RAM.

8.3 Node-type sources

The first class of sources we consider for universality results of our hypersuccinct encoding are *node-type sources*. Node-type sources and the related definitions presented below closely resemble label-shape processes and the corresponding related concepts defined Section 6.4. Shortly speaking, node-type sources generate a binary tree $t \in \mathcal{B}^\diamond$ in a top-down way by determining the node type (binary, left-unary, right-unary or leaf) of each node depending on its ancestors' node types.

8.3.1 Higher-order node-type sources

Formally, we make the following definitions. Let $t \in \mathcal{B}^\diamond$ be a binary tree. We define the *type* of a node v as

$$\mathsf{type}(v) = \begin{cases} 0 & \text{if } v \text{ is a leaf,} \\ 1 & \text{if } v \text{ has a single left child (and no right child),} \\ 2 & \text{if } v \text{ is a binary node,} \\ 3 & \text{if } v \text{ has a single right child (and no left child).} \end{cases}$$

For a node v of a binary tree t, we inductively define the *type history* of v, $h^{\mathsf{type}}(v) \in \{1,2,3\}^*$, as follows: If v is the root node, we set $h^{\mathsf{type}}(v) = \epsilon$, (that is, the empty string). If a node v is the child node of a node w of t, we set $h^{\mathsf{type}}(v) = h^{\mathsf{type}}(w)\mathsf{type}(w)$, that is, in order to obtain $h^{\mathsf{type}}(v)$, we concatenate the types of v's ancestors. Note that $\mathsf{type}(v)$ is not part of the type history of v. Moreover, we define the k-type history of v, $h_k^{\mathsf{type}}(v)$, as the length-k-suffix of $1^k h^{\mathsf{type}}(v)$, that is, if $|h^{\mathsf{type}}(v)| \geq k$, $h_k^{\mathsf{type}}(v)$ equals the last k characters of $h^{\mathsf{type}}(v)$, and if $|h^{\mathsf{type}}(v)| < k$, we pad this too short history with 1's in order to obtain a string $h_k^{\mathsf{type}}(v)$ of length k.[1] Let $k \geq 0$, let $z \in \{1,2,3\}^k$ and let $i \in \{0,1,2,3\}$. With $\hat{n}_z^t$ we denote the number of nodes of t with k-type history z and with $\hat{n}_{z,i}^t$ we denote the number of nodes of type i and k-type history z.

A *k^{th}-order node-type process* $\mathcal{P} = (P_z)_{z \in \{1,2,3\}^k}$ is a tuple of probability mass functions $P_z \colon \{0,1,2,3\} \to [0,1]$. A k^{th} order node-type process assigns a probability $\mathsf{Prob}_{\mathcal{P}}(t)$ to a binary tree $t \in \mathcal{B}^\diamond$ by

$$\mathsf{Prob}_{\mathcal{P}}(t) = \prod_{v \in V(t)} P_{h_k^{\mathsf{type}}(v)}(\mathsf{type}(v)) = \prod_{z \in \{1,2,3\}^k} \prod_{i=0}^{3} (P_z(i))^{\hat{n}_{z,i}^t}. \tag{8.4}$$

We also use the term *node-type source* for a k^{th}-order node-type process. A k^{th}-order node-type source randomly generates a binary tree $t \in \mathcal{B}^\diamond$ as follows: In a top-down way, starting at the root node, we determine for each node v its type, where this decision depends on the k-type history $h_k^{\mathsf{type}}(v)$ of the node. The probability that a node v is of type i is given by $P_{h_k^{\mathsf{type}}(v)}(i)$. If $i = 0$, then

[1] As in the case of label-shape histories (Section 6.4) and label-histories (Section 7.2), several alternatives of treating nodes with shorter type history are possible.

this node becomes a leaf and the process stops at this node. If $i = 1$, we attach a single left child to the node, if $i = 2$, we attach a left and a right child to the node, and if $i = 3$, we attach a single right child to the node. The process then continues at these child nodes. Note that this process might produce infinite trees with non-zero probability.

In the same way as label-shape processes (defined in Section 6.4) correspond to label-shape entropy, node-type sources naturally induce a notion of empirical entropy for binary trees. We define the following higher-order empirical entropy for binary trees:

Definition 8.6 (Node-type entropy)**.** Let $k \geq 0$ be an integer, and let $t \in \mathcal{B}^\diamond$ be a binary tree. The (unnormalized) *k^{th}-order node type entropy* $H_k^{\mathsf{type}}(t)$ of t is defined as

$$H_k^{\mathsf{type}}(t) = \sum_{z \in \{1,2,3\}^k} \sum_{i=0}^{3} \hat{n}_{z,i}^t \log\left(\frac{\hat{n}_z^t}{\hat{n}_{z,i}^t}\right).$$

Again, we make the convention that $0\log(0) = 0$ and $0\log(x/0) = 0$ for $x \geq 0$. The corresponding normalized entropy measure is obtained by dividing by the tree size. The zeroth order empirical type entropy is a slight variant of the degree entropy defined for plane trees in [60] (see Definition 7.2) and occurs implicitly in [21]. Note that node-type entropy is only reasonable in the particular setting of binary trees $t \in \mathcal{B}^\diamond$, thus, it is not included in the entropy comparison of Chapter 7. The k^{th}-order node-type entropy is a lower bound on the coding length of a tree encoder that encodes the type of each node, depending on the k-type history of the node. Normalized node-type entropy measures the expected uncertainty about the type of a node, given the types of the node's k closest ancestors.

We say that the k^{th}-order node-type process $(P_z)_{z\in\{1,2,3\}^k}$ is the *empirical k^{th}-order node-type process of a tree t*, if $P_z(i) = \frac{\hat{n}_{z,i}^t}{\hat{n}_z^t}$ for all $z \in \{1,2,3\}^k$ with $\hat{n}_z^t > 0$ and $i \in \{0,1,2,3\}$. In particular, if $\mathcal{P} = (P_z)_{z\in\{1,2,3\}^k}$ is the empirical k^{th}-order node-type process of a binary tree $t \in \mathcal{B}^\diamond$, we have

$$\begin{aligned}\log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) &= \sum_{z\in\{1,2,3\}^k} \sum_{i=0}^{3} \hat{n}_{z,i}^t \log\left(\frac{1}{P_z(i)}\right) = \sum_{z\in\{1,2,3\}^k} \sum_{i=0}^{3} \hat{n}_{z,i}^t \log\left(\frac{\hat{n}_z^t}{\hat{n}_{z,i}^t}\right) \\ &= H_k^{\mathsf{type}}(t).\end{aligned}$$

This shows that the node-type entropy is precisely the number of bits an optimal code can achieve for this source. In particular, we obtain the following theorem.

Theorem 8.7. *Let $t \in \mathcal{B}^\diamond$ be a binary tree. For every k^{th}-order node-type process $\mathcal{P}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$, we have*

$$H_k^{\mathsf{type}}(t) \leq -\log \mathsf{Prob}_{\mathcal{P}}(t),$$

with equality if and only if $\mathcal{P}$ is the empirical node-type process of t.

Theorem 8.7 is the analogue of Theorem 6.18 transferred from the setting of label-shape processes to the setting of node-type processes (and hence, also an analogue of a result shown by Gagie in [41] in the setting of empirical string entropy and k^{th}-order Markov sources). The proof of Theorem 8.7 follows precisely the same steps as the proof of Theorem 6.18.

Example 8.8. (Binary trees.) In order to encode a (uniformly random) binary tree $t \in \mathcal{B}_n^\diamond$ of size n, $2n$ bits are necessary [30]. Let $\mathcal{P}$ denote the zeroth-order node-type process defined by $P(0) = P(1) = P(2) = P(3) = \frac{1}{4}$, then for every binary tree $t \in \mathcal{B}^\diamond$ of size n we have $\mathsf{Prob}_{\mathcal{P}}(t) = 4^{-n}$ and in particular, $\log(1/\mathsf{Prob}_{\mathcal{P}}(t)) = 2n$.

Example 8.9 (Full binary trees)**.** Node-type processes $\mathcal{P} = (P_z)_{z\in\{1,2,3\}^k}$ with $P_z(1) = P_z(3) = 0$ for all $z \in \{1,2,3\}^k$ generate full binary trees. Recall that every full binary tree consists of an odd number $n = 2j+1$ of nodes, j binary nodes and $j+1$ leaves for some integer j. If $\mathcal{P}$ is a zeroth-order node-type process, we thus have $\mathsf{Prob}_{\mathcal{P}}(t) = P(0)^{j+1}P(2)^j$ for every $t \in \mathcal{B}_n$. Setting $P(0) = P(2) = \frac{1}{2}$ yields

$$\log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) = (j+1)\log(2) + j\log(2) = n,$$

and $n = 2j+1$ is the minimum number of bits needed to represent a (uniformly chosen) full binary tree $t \in \mathcal{B}_n$.

Example 8.10 (Unary paths)**.** Node-type processes $(P_z)_{z\in\{1,2,3\}^k}$ which satisfy $P_z(2) = 0$ generate unary-path trees, that is, trees only consisting of unary nodes and one leaf. In order to encode a unary-path tree of size $n+1$, we need n bits (to encode the n "directions" left/right). For a fixed integer n, let $\mathcal{P}^n$ denote the zeroth-order node-type process with $P^n(1) = P^n(3) = 1/(2+\varepsilon_n)$ and $P^n(0) = \varepsilon_n/(2+\varepsilon_n)$, for $\varepsilon_n = 2/n$. We have

$$\begin{aligned}\log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}^n}(t)}\right) &= n\log(2+\varepsilon_n) + \log\Big(1+\frac{2}{\varepsilon_n}\Big) = n\log\Big(2+\frac{2}{n}\Big) + \log(n+1)\\ &\le n + \log(n+1) + \frac{1}{\ln(2)},\end{aligned}$$

for every unary-path tree $t \in \mathcal{B}_{n+1}^\diamond$.

Example 8.11 (Motzkin trees)**.** Motzkin trees are binary trees with only one type of unary nodes, see Example 4.3 in Chapter 4. Motzkin trees are generated by node-type processes $(P_z)_{z\in\{1,2,3\}^k}$ with $P_z(3) = 0$ for every $z \in \{1,2,3\}^k$ (and then left-unary nodes are simply treated as unary nodes). For encoding (uniformly random) Motzkin trees of size n, asymptotically $\log(3)n \approx 1.58496n$ bits are necessary [101, Theorem 6.16] (note that this also follows from Theorem 4.6 applied to the simply generated family of Motzkin trees). Let $\mathcal{P}$ denote the zeroth-order node-type process with $P(0) = P(1) = P(2) = \frac{1}{3}$, then $\mathsf{Prob}_{\mathcal{P}}(t) = 3^{-n}$ for every Motzkin tree of size n. In particular, we have $\log(1/\mathsf{Prob}_{\mathcal{P}}(t)) = \log(3)n$.

Example 8.12. In [50], it was shown that the average numbers of node types in a random binary search tree (i.e., a random tree generated according to the binary search tree model, see Definition 2.15) of size n satisfy

$$\sum_{t\in\mathcal{B}_n^\diamond} P_{bst}(t)\cdot\hat{n}_0^t \sim \sum_{t\in\mathcal{B}_n^\diamond} P_{bst}(t)\cdot\hat{n}_2^t \sim \frac{1}{3}n$$

and

$$\sum_{t\in\mathcal{B}_n^\diamond} P_{bst}(t)\cdot\hat{n}_1^t = \sum_{t\in\mathcal{B}_n^\diamond} P_{bst}(t)\cdot n_3^t \sim \frac{1}{6}n.$$

Thus, it seems natural to consider the zeroth-order node-type process $\mathcal{P}$ given by $P(0)=P(2)=\frac{1}{3}$ and $P(1)=P(3)=\frac{1}{6}$. In [21], a data structure supporting RMQ in constant time using

$$\sum_{t\in\mathcal{B}_n^\diamond} P_{bst}(t)\log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right)+o(n)\approx 1.919n+o(n)$$

many bits in expectation is introduced. However, to achieve the asymptotically optimal $\approx 1.736n+o(n)$ bits on average (see [50], respectively, [66] for the entropy rate of the binary search tree model), it is necessary to consider a different kind of binary-tree sources.

8.3.2 Universality with respect to node-type sources

In order to show a universality result of the hypersuccinct code from Section 8.2 with respect to the class of node-type sources, we first derive a source-specific encoding (called a *depth-first arithmetic code*) with respect to the node-type source, against which we will then compare our hypersuccinct code.

The formula for $\mathsf{Prob}_{\mathcal{P}}(t)$, Equation (8.4), suggests a route for an (essentially) optimal *source-specific* encoding of any binary tree $t\in\mathcal{B}^\diamond$ that, given a k^{th}-order node-type process $\mathcal{P}=(P_z)_{z\in\{1,2,3\}^k}$, spends $\log(1/\mathsf{Prob}_{\mathcal{P}}(t))$ (plus lower-order terms) many bits in order to encode a binary tree $t\in\mathcal{B}^\diamond$ with $\mathsf{Prob}_{\mathcal{P}}(t)>0$. Such an encoding may spend $\log(1/P_z(i))$ many bits per node v of type i and of k-type history z of t. (Note that if $\mathsf{Prob}_{\mathcal{P}}(t)>0$ by assumption, we have $P_{h_k^{\mathsf{type}}(v)}(\mathsf{type}(v))>0$ for every node v of t).

Assuming that we "know" the k^{th}-order type process $\mathcal{P}$ – i.e., that it need not be stored as part of the encoding – we can use *arithmetic coding* in order to encode the type of node v in that many bits. A simple (source-dependent) encoding $D_{\mathcal{P}}$, dependent on a given k^{th}-order type process $\mathcal{P}$, thus stores a tree t as follows.

While traversing the tree in depth-first order, we always know the k-type history of each node v we pass, and encode $\mathsf{type}(v)$ of each node v, using arithmetic coding. To encode $\mathsf{type}(v)$, we feed the arithmetic coder with the model that the next symbol is a number $i\in\{0,1,2,3\}$ with probability $P_z(i)$,

where z is the k-type history of v. We refer to this (source-dependent) code $D_{\mathcal{P}}$ as the *depth-first arithmetic code* for the node-type source $\mathcal{P}$.

We can reconstruct the tree t recursively from its code $D_{\mathcal{P}}(t)$, as we always know the node types of nodes we have already visited in the depth-first order traversal of the tree, and the k-type history of the node which we will visit next. As arithmetic coding needs $\log\bigl(1/P_{h_k^{\mathsf{type}}(v)}(\mathsf{type}(v))\bigr)$ many bits per node v, plus at most 2 bits of overhead, the total number of bits needed to store a binary tree $t \in \mathcal{B}^{\diamond}$ is thus

$$|D_{\mathcal{P}}(t)| \leq \sum_{v \in V(t)} \log\left(\frac{1}{P_{h_k^{\mathsf{type}}(v)}(\mathsf{type}(v))}\right) + 2 = \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + 2. \tag{8.5}$$

Note that $D_{\mathcal{P}}$ is a single prefix-free code for the set of all binary trees t which satisfy $\mathsf{Prob}_{\mathcal{P}}(t) > 0$ with respect to the k^{th}-order type process $\mathcal{P} = (P_z)_{z \in \{1,2,3\}^k}$. We start with the following lemma:

Lemma 8.13. *Let $\mathcal{P} = (P_z)_{z \in \{1,2,3\}^k}$ be a k^{th}-order type process and let $t \in \mathcal{B}_n^{\diamond}$ be a binary tree of size n with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. Then*

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}\left(\frac{nk + n\log\log n}{\log n}\right),$$

where Ψ is a Huffman code for the sequence of the micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ obtained from the tree covering scheme (see Section 8.2).

Proof. Let v be a node of t and let $\mathfrak{t}_i$ denote the micro tree of t that contains v. For the sake of clarity, let $\mathsf{type}^t(v)$ denote the type of v viewed as a node of t, and let $\mathsf{type}^{\mathfrak{t}_i}(v)$ denote the type of v in $\mathfrak{t}_i$. We find that $\mathsf{type}^{\mathfrak{t}_i}(v) = \mathsf{type}^t(v)$, unless v is a parent of a portal. In this case, the degree of v in $\mathfrak{t}_i$ is strictly smaller than the degree of v in t.

By definition of the tree covering scheme (Lemma 8.2), there are at most two parents of portal nulls per micro tree $\mathfrak{t}_i$. If a tree $\mathfrak{t}_i$ contains two parents of portal nulls, one of those two nodes is the root node by Lemma 8.2. Let $\pi_{i,1}$ denote the root node of $\mathfrak{t}_i$ and let $\pi_{i,2}$ denote the parent node of the portal null in $\mathfrak{t}_i$ which is not the root node, if it exists. Moreover, let $\mathrm{pos}(\pi_{i,2})$ denote the preorder index of node $\pi_{i,2}$ in $\mathfrak{t}_i$.

Again for the sake of clarity, let $h_k^{\mathsf{type},t}(v)$ denote the k-type history of v in t, and let $h_k^{\mathsf{type},\mathfrak{t}_i}(v)$ denote the k-type history of v in micro tree $\mathfrak{t}_i$. If v does not have k ancestors within $\mathfrak{t}_i$, then its k-type history $h_k^{\mathsf{type},\mathfrak{t}_i}(v)$ in $\mathfrak{t}_i$ might not coincide with its k-type history $h_k^{\mathsf{type},t}(v)$ in t, and if v is a descendant of order smaller than k of node $\pi_{i,2}$, then its k-type history in $\mathfrak{t}_i$ does not coincide with its k-type history in t, as $\pi_{i,2}$ changes its node type.

However, if we know the k-type history $h_k^{\mathsf{type},t}(\pi_{i,1})$ of the root node $\pi_{i,1}$ of $\mathfrak{t}_i$, the type $\mathsf{type}^t(\pi_{i,1})$, and the preorder position (in $\mathfrak{t}_i$) and type (in t) of the node $\pi_{i,2}$, we are able to recover the k-type history $h_k^{\mathsf{type},t}(v)$ of every node $v \in \mathfrak{t}_i$. We define the following modification of $D_{\mathcal{P}}$ (the depth-first arithmetic code defined

at the beginning of this section), under the assumption that we know $h_k^{\mathsf{type},t}(\pi_{i,1})$, $\mathsf{type}^t(\pi_{i,1})$, $\mathsf{type}^t(\pi_{i,2})$ and $\mathrm{pos}(\pi_{i,2})$.

While traversing the micro-tree t_i in depth-first order, we encode $\mathsf{type}^{\mathsf{t}_i}(v)$ (that is, $\mathsf{type}^t(v)$) for every node v of t_i except for nodes $\pi_{i,1}$ and $\pi_{i,2}$ (if it exists), for which we encode $\mathsf{type}^t(\pi_{i,1})$ and $\mathsf{type}^t(\pi_{i,2})$ (which we know, by assumption, as well as the preorder position of $\pi_{i,2}$); as we know $h_k^{\mathsf{type},t}(\pi_{i,1})$ by assumption, as well as the node types of $\pi_{i,1}$ and $\pi_{i,2}$, we know $h_k^{\mathsf{type},t}(v)$ at every node v we pass. We therefore encode $\mathsf{type}^t(v)$ using arithmetic coding by feeding the arithmetic coder with the model that the next symbol is a number $i \in \{0,1,2,3\}$ with probability $P_{h_k^{\mathsf{type},t}(v)}(i)$.

Let $\tilde{z} = h_k^{\mathsf{type},t}(v)$. We denote this modification of $D_{\mathcal{P}}(\mathsf{t}_i)$ with $D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i)$ and find that it spends at most

$$|D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i)| \leq \sum_{v \in V(\mathsf{t}_i)} \log\Big(\frac{1}{P_{h_k^{\mathsf{type},t}(v)}(\mathsf{type}^t(v))}\Big) + 2 \tag{8.6}$$

many bits in order to encode a micro tree t_i.

Furthermore, let $\Delta\colon \{0,1,2,3\}^\star \to \{0,1\}^*$ denote any uniquely decodable binary encoding which spends $2|z|$ bits in order to encode $z \in \{0,1,2,3\}^*$. Let $\mathcal{I}_0$ denote the set of indices $i \in \{1,\ldots,m\}$ for which t_i is a fringe subtree of t (that is, t_i does not contain any portal nulls), let $\mathcal{I}_1$ denote the set of indices $i \in \{1,\ldots,m\}$ for which the root node of t_i is a parent of a portal null, but no other portal null exists, let $\mathcal{I}_3$ denote the set of indices $i \in \{1,\ldots,m\}$, for which the root node of t_i is not a parent of a portal null, but node $\pi_{i,2}$ is a parent of a portal null, and let $\mathcal{I}_3 = \{1,\ldots,m\} \setminus (\mathcal{I}_0 \cup \mathcal{I}_1 \cup \mathcal{I}_2)$. We define a modified encoding of t_i as follows.

$$\tilde{D}_{\mathcal{P}}(\mathsf{t}_i) = \begin{cases} 00 \cdot \Delta(h_k^{\mathsf{type},t}(\pi_{i,1})) \cdot D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i), & \text{if } i \in \mathcal{I}_0; \\ 01 \cdot \Delta(h_k^{\mathsf{type},t}(\pi_{i,1})) \cdot \gamma(\mathsf{type}^t(\pi_{i,1})+1) \cdot D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i), & \text{if } i \in \mathcal{I}_1; \\ 10 \cdot \Delta(h_k^{\mathsf{type},t}(\pi_{i,1})) \cdot \gamma(\mathrm{pos}(\pi_{i,2})) \cdot \gamma(\mathsf{type}^t(\pi_{i,2}+1)) & \\ \quad \cdot D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i), & \text{if } i \in \mathcal{I}_2; \\ 11 \cdot \Delta(h_k^{\mathsf{type},t}(\pi_{i,1})) \cdot \gamma(\mathsf{type}^t(\pi_{i,1}+1)) \cdot \gamma(\mathrm{pos}(\pi_{i,2})) & \\ \quad \cdot \gamma(\mathsf{type}^t(\pi_{i,2})+1) \cdot D_{\mathcal{P},\tilde{z}}(\mathsf{t}_i) & \text{if } i \in \mathcal{I}_3. \end{cases}$$

Here, γ again denotes the Elias gamma encoding of positive integers.

Note that formally, $\tilde{D}_{\mathcal{P}}$ is *not* a prefix-free code over $\Sigma_{s_{\max}}$, as there can be micro tree shapes that are assigned several codewords by $\tilde{D}_{\mathcal{P}}$, depending on which and how many nodes are portals to other micro trees. Nevertheless, $\tilde{D}_{\mathcal{P}}$ is uniquely decodable to local shapes of micro trees, and can thus be seen as a *generalized prefix-free code*, where more than one codeword per symbol is allowed.

In terms of encoding length, assigning more than one codeword is not helpful – removing all but the shortest one never makes the code worse – so a Huffman

code minimizes the encoding length over the larger class of *generalized* prefix-free codes. We find

$$\begin{aligned}\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| &\leq \sum_{i=1}^{m} |\tilde{D}_{\mathcal{P}}(\mathfrak{t}_i)| = \sum_{j=0}^{3} \sum_{i \in \mathcal{I}_j} |\tilde{D}_{\mathcal{P}}(\mathfrak{t}_i)| \\ &\leq \sum_{i=1}^{m} |\Delta(h_k^{\mathsf{type},t}(\pi_{i,1}))| + \sum_{j=0}^{3} \sum_{i \in \mathcal{I}_j} |D_{\mathcal{P},\tilde{z}}(\mathfrak{t}_i)| + \mathcal{O}(m \log s_{\max}),\end{aligned}$$

as $\mathrm{pos}(\pi_{i,2}) \leq s_{\max}$. With the estimate (8.6), we find

$$\begin{aligned}\sum_{j=0}^{3} \sum_{i \in \mathcal{I}_j} |D_{\mathcal{P},\tilde{z}}(\mathfrak{t}_i)| &\leq \sum_{j=0}^{3} \sum_{i \in \mathcal{I}_j} \sum_{v \in V(\mathfrak{t}_i)} \log\Big(\frac{1}{P_{h_k^{\mathsf{type},t}(v)}(\mathsf{type}^t(v))}\Big) + 2m \\ &= \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + 2m.\end{aligned}$$

Hence, as $|\Delta(h_k^{\mathsf{type},t}(\pi_{i,1}))| = 2k$, we have

$$\begin{aligned}\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| &\leq \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}(m \log s_{\max}) + \mathcal{O}(km) \\ &= \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}\left(\frac{nk + n \log\log n}{\log n}\right),\end{aligned}$$

as $m = \Theta(n/\log n)$ and $s_{\max} = \Theta(\log n)$ (see Section 8.2). This finishes the proof of the lemma. □

From Lemma 8.13 and Lemma 8.5, we obtain the following result.

Theorem 8.14. *Let $\mathcal{P}$ be a k^{th}-order node-type process. The hypersuccinct encoding $\mathsf{H}\colon \mathcal{B}^{\diamond} \to \{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)| \leq \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}\left(\frac{nk + n \log\log n}{\log n}\right)$$

for every $t \in \mathcal{B}_n^{\diamond}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. In particular, if $\mathcal{P}$ is the empirical k^{th}-order node-type process of the binary tree t, we obtain the entropy bound

$$|\mathsf{H}(t)| \leq H_k^{\mathsf{type}}(t) + \mathcal{O}\left(\frac{nk + n \log\log n}{\log n}\right).$$

By Theorem 8.7, we find that the k^{th}-order node-type entropy $H_k^{\mathsf{type}}(t)$ lower-bounds the self-information $-\log \mathsf{Prob}_{\mathcal{P}}(t)$ over all k^{th}-order node-type sources $\mathcal{P}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. Note that the redundancy terms in Theorem 8.14 are identical to the redundancy terms in the universality result from Theorem 6.20 and the entropy bound from Theorem 6.21 from Chapter 6 (for the special case of unlabeled full binary trees). Moreover, note that the redundancy term only becomes a lower-order term if $k \leq o(\log n)$; the same assumption on k can be

found in Theorem 6.21 for the label-shape entropy, respectively, the entropy bounds from [89, 46].

From Theorem 8.14 and Example 8.8, Example 8.9, Example 8.10 and Example 8.11 we obtain the following corollary.

Corollary 8.15. *The hypersuccinct code* $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ *compresses*

- ***binary trees*** *t of size n, drawn uniformly at random from the set $\mathcal{B}_n^\diamond$ of all binary trees of size n, using $|\mathsf{H}(t)| \leq 2n + \mathcal{O}(n \log\log n / \log n)$ many bits,*

- ***full binary trees*** *t of size n, drawn uniformly at random from the set $\mathcal{B}_n$ of all full binary trees of size n, using $|\mathsf{H}(t)| \leq n + \mathcal{O}(n \log\log n / \log n)$ many bits,*

- ***unary-path trees*** *t of size $n+1$, drawn uniformly at random from the set of all unary-path trees of size $n+1$, using $|\mathsf{H}(t)| \leq n + \mathcal{O}(n \log\log n / \log n)$ many bits, and*

- ***Motzkin trees*** *t of size n, drawn uniformly at random from the set of all Motzkin trees of size n, using $|\mathsf{H}(t)| \leq \log(3)n + \mathcal{O}(n \log\log n / \log n)$ many bits.*

8.4 Fixed-size and fixed-height binary tree sources

In this section, we show that our hypersuccinct encoding satisfies universality results with respect to many classes of leaf-centric and depth-centric binary tree sources, with respect to which the encodings E_{dag} and E_{tslp} from [105], [S3] are shown to be universal (see Section 8.1).

However, in [105], [S3], the family of *full binary trees* $\mathcal{B}$ is considered, whereas our hypersuccinct encoding is defined for the more general case of the family of binary trees $\mathcal{B}^\diamond$. Recall from Chapter 2, that the families $\mathcal{B}$ and $\mathcal{B}^\diamond$ are in a natural one-to-one correspondence; thus, every leaf-centric binary tree source and every depth-centric binary tree source naturally induces a corresponding tree source for the family of trees $\mathcal{B}^\diamond$. Thus, we consider these analogues of leaf-centric and depth-centric binary tree sources in this chapter, which are called *fixed-size binary tree sources* and *fixed-height binary tree sources*.

A fixed-size binary tree source induces a probability mass function on the family of binary trees $\mathcal{B}_n^\diamond$ for every $n \in \mathbb{N}$; this corresponds to the concept of leaf-centric binary tree sources (as considered in Chapter 3 of this work). A fixed-height binary tree source induces a probability mass function on the family of binary trees of height n for every $n \in \mathbb{N}$; this is the analogue of depth-centric binary tree sources.

8.4.1 Fixed-size binary tree sources

A fixed-size binary tree source is defined by a function $\ell\colon \mathbb{N}_0^2 \to [0,1]$, such that

$$\sum_{i=0}^{n} \ell(i, n-i) = 1 \qquad \text{for all } n \in \mathbb{N}_0.$$

A fixed-size binary tree source ℓ induces a probability mass function over the set of all binary trees $\mathcal{B}_n^\diamond$ of size n for every $n \geq 1$ by

$$P_\ell(t) = \prod_{v \in V(t)} \ell(|t_l[v]|, |t_r[v]|). \tag{8.7}$$

Note that $t_l[v]$ or $t_r[v]$ can be the empty tree of size zero. If t is the empty tree, we set $P_\ell(t) = 1$.

Intuitively, this corresponds to generating a binary tree by a (recursive) depth-first traversal as follows. Given a target tree size n, ask the source for a left subtree size $i \in \{0, ..., n-1\}$. The probability of a left subtree size i is $\ell(i, n-1-i)$. Create a node and recursively generate its left subtree of size i and its right subtree of size $n-1-i$. The random choices in the left and right subtree are independent conditional on their sizes. An inductive proof over n verifies that $\sum_{t \in \mathcal{B}_n^\diamond} P_\ell(t) = 1$ for every $n \in \mathbb{N}_0$.

Note that every fixed-size binary-tree source naturally corresponds to a leaf-centric binary-tree source (as defined in Definition 3.1) and vice-versa. In particular, the leaf-centric binary tree sources considered in Example 3.2, Example 3.3 and Example 3.4 have their analogues as fixed-size binary tree sources.

Example 8.16 (Random binary search tree model)**.** Naturally, the *binary search tree model* (defined in Definition 2.15) corresponds to a fixed-size binary tree source ℓ_{bst} with $\ell_{bst}(i, n-i) = \frac{1}{n+1}$ for all $i \in \{0, \ldots, n\}$ and $n \in \mathbb{N}_0$. The corresponding leaf-centric binary tree source is given in Example 3.2.

Example 8.17 (Uniform distribution)**.** Also the *uniform probability distribution* (Definition 2.14) corresponds to a fixed-size tree source, which is defined by

$$\ell_{uni}(i, n-i) = \frac{|\mathcal{B}_i^\diamond||\mathcal{B}_{n-i}^\diamond|}{|\mathcal{B}_{n+1}^\diamond|} \qquad \text{for every } i \in \{0, \ldots, n-1\} \text{ and } n \in \mathbb{N}_0.$$

This corresponds to the leaf-centric binary tree source from Example 3.3.

Example 8.18 (Binomial random tree model)**.** Fix a constant $0 < p < 1$ and recall the *binomial random tree model* from Example 3.4. The corresponding fixed-size binary tree source is defined by

$$\ell_{bin,p}(i, n-i) = p^i (1-p)^{n-i} \binom{n}{i}$$

for every $i \in \{0, \ldots, n\}$ and $n \in \mathbb{N}_0$.

Example 8.19 (Almost paths)**.** Setting $\ell(0,n) = \ell(n,0) = \frac{1}{2}$ for $n \geq 2$ yields a fixed-size binary tree source which produces unary paths; (this is a special case of [105, Ex. 6]). One can generalize the example so that $\ell(i,j) > 0$ implies $\min\{i,j\} \leq K$ for some constant K by setting

$$\ell_{path}(i,j) = \begin{cases} \min\left\{\frac{1}{j+i+1}, \frac{1}{2(K+1)}\right\} & \text{if } i \leq K \text{ or } j \leq K, \\ 0 & \text{otherwise.} \end{cases}$$

A fixed-size source ℓ_{path} only generates binary trees for which at each node, the left or right subtree has at most K nodes. Unary paths correspond to $K = 0$.

Example 8.20 (Random fringe-balanced binary search tree model)**.** Let $\alpha \in \mathbb{N}_0$ be a parameter, and define

$$\ell_{bal}(k, n-k-1) = \begin{cases} \binom{k}{\alpha} \cdot \binom{n-k-1}{\alpha} \cdot \binom{n}{2\alpha+1}^{-1} & \text{if } n \geq 2\alpha+1, \\ \frac{1}{n} & \text{otherwise.} \end{cases}$$

This yields random $(2\alpha+1)$-fringe-balanced binary search trees; (see [103, §4.3] and the references therein for background on these trees).

8.4.2 Fixed-height binary-tree sources

Recall that $\hbar(t)$ denotes the height of a tree (which is the depth of the tree plus one), and that we defined the height of the empty tree to be zero. A fixed-height binary tree source is defined by a function $\ell\colon \mathbb{N}_0^2 \to [0,1]$, such that

$$\sum_{\substack{i,j \in \mathbb{N}_0 \\ \max(i,j)=n}} \ell(i,j) = 1 \qquad \text{for all } n \in \mathbb{N}_0.$$

A fixed-height tree source induces a probability mass function over the set of all binary trees $t \in \mathcal{B}^\diamond$ of height n by

$$P_\ell(t) = \prod_{v \in V(t)} \ell(\hbar(t_l[v]), \hbar(t_r[v])). \tag{8.8}$$

Note again that $t_l[v]$ or $t_r[v]$ can be the empty tree of height zero. If t is the empty tree, we set $P_\ell(t) = 1$.

Intuitively, this corresponds to generating a binary tree by a (recursive) depth-first traversal as follows: Given a target height n of the tree, ask the source for the height i of the left subtree and the height j of the right subtree conditional on $\max(i,j) = n-1$. The probability of a pair of heights (i,j) with $\max(i,j) = n-1$ is $\ell(i,j)$. Create a node and recursively generate its left subtree of height i and its right subtree of height j. The random choices in the left and right subtree are again independent conditional on their heights.

An inductive proof over n verifies that $\sum_t P_\ell(t) = 1$ for every $n \in \mathbb{N}_0$, where the sum is taken over all binary trees $t \in \mathcal{B}^\diamond$ of height n. The concept of fixed-height binary-tree sources corresponds to the concept of depth-centric binary-tree sources considered in [105], [S3].

Example 8.21 (AVL trees by height)**.** An AVL tree is a binary tree $t \in \mathcal{B}^\diamond$, such that for every node v of t, we have $|\hbar(t_l[v]) - \hbar(t_r[v])| \leq 1$. Let $\mathcal{A}^n \subseteq \mathcal{B}^\diamond$ denote the set of AVL trees of height n. The number of AVL trees of height n satisfies the following recurrence relation:

$$|\mathcal{A}^n| = 2|\mathcal{A}^{n-1}||\mathcal{A}^{n-2}| + |\mathcal{A}^{n-1}||\mathcal{A}^{n-1}|.$$

Set

$$\ell_{avl}(i,j) = \begin{cases} \dfrac{|\mathcal{A}^i||\mathcal{A}^j|}{|\mathcal{A}^n|} & \text{for } (i,j) \in \{(n-2,n-1),(n-1,n-1),(n-1,n-2)\} \\ 0 & \text{otherwise,} \end{cases}$$

for every $n \geq 2$. Then ℓ_{avl} corresponds to a uniform probability distribution on the set $\mathcal{A}^n$ of AVL trees of *height* n for every $n \in \mathbb{N}$.

8.4.3 Monotonic fixed-size and fixed-height sources

In the following, we present several classes of fixed-size and fixed-height binary-tree sources, for which we will be able to show that our hypersuccinct encoding is universal. The first property (respectively, the analogous property for leaf-centric and depth-centric binary tree sources) was introduced in [S3].

Definition 8.22 (Monotonic source)**.** A fixed-size or fixed-height binary tree source is *monotonic* if $\ell(i,j) \geq \ell(i+1,j)$ and $\ell(i,j) \geq \ell(i,j+1)$ for all $i,j \in \mathbb{N}_0$.

Specifically, it is shown in [S3], that the tree encoder E_{tslp} based on TSLP compression is universal with respect to the classes of monotonic leaf-centric and monotonic depth-centric tree sources in the sense that the worst-case redundancy (8.1) of E_{tslp} converges to zero for every monotonic leaf- or depth-centric source.[2]

Clearly, the binary search tree model ℓ_{bst} from Example 8.16 is a monotonic fixed-size tree source, and one can easily show that the uniform model ℓ_{uni} from Example 8.17 is another one. Furthermore, the fixed-size source ℓ_{path} from Example 8.19 is monotonic. In contrast, the binomial random tree model $\ell_{bin,p}$ from Example 8.18 and the fringe-balanced binary search tree model (Example 8.20) are not monotonic.

For monotonic tree sources, we find the following:

[2] To be more precise, in [S3] it is shown that E_{tslp} is worst-case universal with respect to all leaf- and depth-centric binary tree sources that satisfy a property called *strong domination property*. Monotonic leaf- and depth-centric binary tree sources are shown to satisfy the strong-domination property. It is an open problem whether there are leaf- or depth-centric tree sources which satisfy the strong-domination property and which are not monotonic.

Lemma 8.23. *Let $t \in \mathcal{B}^\diamond$, and let $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ be a partition of t into disjoint subtrees, in the sense that every node of t belongs to exactly one subtree $\mathfrak{t}_i$. If ℓ is a monotonic fixed-size or monotonic fixed-height binary tree source, then*

$$P_\ell(t) \leq \prod_{i=1}^{m} P_\ell(\mathfrak{t}_i).$$

Proof. Let v be a node of t and let $\mathfrak{t}_i$ denote the subtree that v belongs to. As $\mathfrak{t}_i$ is a (not necessarily fringe) subtree of t, we find $|\mathfrak{t}_{il}[v]| \leq |t_l[v]|$, $|\mathfrak{t}_{ir}[v]| \leq |t_r[v]|$ as well as $\hbar(\mathfrak{t}_{il}[v]) \leq \hbar(t_l[v])$ and $\hbar(\mathfrak{t}_{ir}[v]) \leq \hbar(t_r[v])$. From the definition of monotonicity, we thus have $\ell(|\mathfrak{t}_{il}[v]|, |\mathfrak{t}_{ir}[v]|) \geq \ell(|t_l[v]|, |t_r[v]|)$, if ℓ corresponds to a fixed-size source, respectively, $\ell(\hbar(\mathfrak{t}_{il}[v]), \hbar(\mathfrak{t}_{ir}[v])) \geq \ell(\hbar(t_l[v]), \hbar(t_r[v]))$, if ℓ corresponds to a fixed-height source. As every node of t belongs to exactly one subtree $\mathfrak{t}_i$, we find for monotonic fixed-size sources ℓ:

$$P_\ell(t) = \prod_{v \in V(t)} \ell\big(|t_l[v]|, |t_r[v]|\big) \leq \prod_{i=1}^{m} \prod_{v \in V(\mathfrak{t}_i)} \ell\big(|\mathfrak{t}_{il}[v]|, |\mathfrak{t}_{ir}[v]|\big) = \prod_{i=1}^{m} P_\ell(\mathfrak{t}_i).$$

For monotonic fixed-height sources, we similarly find

$$P_\ell(t) = \prod_{v \in V(t)} \ell\big(\hbar(t_l[v]), \hbar(t_r[v])\big) \leq \prod_{i=1}^{m} \prod_{v \in V(\mathfrak{t}_i)} \ell\big(\hbar(\mathfrak{t}_{i\ell}[v]), \hbar(\mathfrak{t}_{ir}[v])\big) = \prod_{i=1}^{m} P_\ell(\mathfrak{t}_i).$$

□

This lemma depicts the crucial property of monotonic sources, based on which we will be able prove universality of our hypersuccinct encoding from Section 8.2 with respect to these classes of fixed-size and fixed-height binary tree sources. The corresponding result and proof will be presented in Section 8.4.5.

8.4.4 Fringe-dominated tree sources

Further classes of fixed-size and fixed-height tree sources for which we will be able to show universality of our encoding are the following. These classes widely overlap with the classes of leaf- and depth-centric binary tree sources, with respect to which the tree encoder E_{dag} from [105] is shown to be (worst-case or average-case) universal. Let $T_{n,\ell}$ again denote a random tree of size n generated by a fixed-size binary tree source ℓ and let $Y_{n,k}$ again denote the (random) number of nodes v in $T_{n,\ell}$, which satisfy $|T_{n,\ell}[v]| > k$. Moreover, let $y_k(t)$ denote the number of nodes v in a tree $t \in \mathcal{B}^\diamond$ which satisfy $|t[v]| > k$. We make the following definitions.

Definition 8.24 (Average-case fringe-dominated)**.** We call a fixed-size binary tree source *average-case κ-fringe dominated* for an increasing function κ with $\kappa(n) = \Theta(\log(n))$, if

$$\mathbb{E}(Y_{n,\kappa(n)}) = o\left(\frac{n}{\log(\kappa(n))}\right).$$

Definition 8.25 (Worst-case fringe-dominated)**.** We call a fixed-size or fixed-height binary tree source *worst-case κ-fringe dominated* for an increasing function κ with $\kappa(n) = \Theta(\log(n))$, if

$$y_{\kappa(n)}(t) = o\left(\frac{n}{\log(\kappa(n))}\right).$$

for every tree $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$.

Note that Definition 8.25 treats fixed-size and fixed-height binary tree sources, but Definition 8.24 only covers fixed-size binary tree sources (to avoid averaging over trees of different sizes). Moreover, a fixed-size tree source that is worst-case κ-fringe-dominated is clearly average-case κ-fringe-dominated as well.

Furthermore, note that the properties of average-case and worst-case fringe-domination are closely related to DAG-compression via the cut-point argument from Section 3.2 (see in particular Lemma 3.5). Recall that the tree encoder E_{dag} from [105], [S3] is shown to be average-case (respectively, worst-case) universal with respect to the classes of leaf-centric and depth-centric tree sources, for which the average (respectively, worst-case) compression ratio achieved by the minimal DAG converges to zero (i.e., universal in the sense that the average-case redundancy (8.2), respectively, worst-case redundancy (8.1) converges to zero). We will show that the classes of leaf-centric and depth-centric tree sources, with respect to which E_{dag} is universal, are widely covered by universality results for our hypersuccinct encoding (see Theorem 8.42 and Theorem 8.43 in Section 8.4.5).

Sufficient conditions for fixed-size sources to be average-case fringe-dominated are given in Chapter 3 in the context of DAG-compression. As every leaf-centric binary tree source induces a fixed-size binary tree source, the results from Chapter 3 apply in a natural way in the setting of binary trees from $\mathcal{B}^\diamond$ as well. The classes of leaf-centric binary tree sources from Chapter 3 correspond the following classes of fixed-size binary tree sources:

Definition 8.26 (ψ-upper-bounded sources, Definition 3.8)**.** Let $\psi\colon \mathbb{R} \to [0,1]$ be a decreasing function. We call a fixed-size source ℓ *ψ-upper-bounded*, if there is an integer N_ℓ such that

$$\ell(i, n-i) \leq \psi(n)$$

for all $n \geq N_\ell$ and $0 \leq i \leq n$.

Definition 8.27 (φ-weakly-balanced sources, Definition 3.14)**.** Let $\varphi\colon \mathbb{N} \to (0,1]$ be a decreasing function and let $\gamma \in (0, \frac{1}{2})$. We call a fixed-size source ℓ *φ-weakly-balanced*, if there is an integer N_ℓ such that

$$\sum_{\gamma n \leq i \leq (1-\gamma)n} \ell(i-1, n-i-1) \geq \varphi(n)$$

for all $n \geq N_\ell$.

Definition 8.28 (ς-strongly-balanced sources, Definition 3.19)**.** Let $\varsigma\colon \mathbb{R} \to (0,1]$ be a decreasing function and let $\gamma \in (0, \frac{1}{2})$. We call a fixed-size source ℓ *ς-strongly-balanced*, if there is an integer N_ℓ and a constant $c_\ell \geq 1$ such that for every $n \geq N_\ell$ and every integer r with $c_\ell \leq r \leq \lceil \gamma n \rceil$, the following inequality holds:

$$\sum_{r \leq i \leq n-r} \ell(i-1, n-i-1) \geq \varsigma(r).$$

In Chapter 3, we have already investigated the expectation $\mathbb{E}(Y_{n,k})$ with respect to leaf-centric binary tree sources from the three classes of sources presented above. We thus obtain the following three lemmas from the corresponding results proven in Chapter 3.

Lemma 8.29 (see Lemma 3.11)**.** *Let ℓ be a ψ-upper-bounded fixed-size binary tree source, then*

$$\mathbb{E}(Y_{n,\kappa(n)}) \leq \mathcal{O}(n\psi(\kappa(n)))$$

for every increasing function κ with $\kappa(n) = \Theta(\log n)$.

Lemma 8.30 (see Lemma 3.16)**.** *Let ℓ be a φ-weakly-balanced fixed-size binary tree source, then*

$$\mathbb{E}(Y_{n,\kappa(n)}) \leq \mathcal{O}\left(\frac{n}{\gamma\varphi(n)\kappa(n)}\right)$$

for every increasing function κ with $\kappa(n) = \Theta(\log n)$.

Lemma 8.31 (see Lemma 3.21)**.** *Let ℓ be a ς-strongly-balanced fixed-size binary tree source, then*

$$\mathbb{E}(Y_{n,\kappa(n)}) \leq \mathcal{O}\left(\frac{n}{\gamma\varsigma(\kappa(n))\kappa(n)}\right)$$

for every increasing function κ with $\kappa(n) = \Theta(\log n)$.

Thus, if a fixed-size tree source ℓ is ψ-upper-bounded for a function ψ with $\psi(n) \leq o(1/\log(n))$, or φ-weakly-balanced for a decreasing function φ with $\varphi(n) \geq \omega(\log\log n/\log n)$, or ς-strongly-balanced for a decreasing function ς with $\varsigma(n) \geq \omega(\log n/n)$, then it is average-case fringe dominated.

The binary search tree model ℓ_{bst} from Example 8.16, is average-case fringe dominated (as we can choose $\psi(n) = \Theta(1/n)$ in Lemma 8.29 and $\varphi(n) = \Theta(1)$ in Lemma 8.30, see also Examples 3.12 and 3.17). Moreover, the binomial random tree model $\ell_{bin,p}$ from Example 8.18, and the uniform model ℓ_{uni} from Example 8.17 are average-case fringe dominated (see Examples 3.18 and 3.22). Additionally, for the random fringe-balanced binary search tree model from Example 8.20, it is easy to show that it is average-case fringe dominated by choosing $\psi(n) = \Theta(1/n)$ in Lemma 8.29 (see also [102, Lemma 2.38]).

Intuitively, φ-weakly-balanced fixed-size tree sources lower-bound the probability of balanced binary trees in terms of the function φ. They generalize a

class of tree sources considered in [S3, Lemma 4 and Theorem 2], as well as leaf-balanced (called weight-balanced below) tree sources introduced in [105] and further analyzed in [S3].

Definition 8.32 (weight-balanced sources, [105], [S3])**.** A weight-balanced fixed-size binary tree source ℓ is a φ-weakly-balanced fixed-size binary tree source with $\varphi = 1$, that is, there is a constant $\gamma \in (0, \frac{1}{2})$, such that

$$\sum_{\gamma n \leq i \leq (1-\gamma)n} \ell(i-1, n-i-1) = 1$$

for every $n \in \mathbb{N}$.

Weight-balanced fixed-size binary tree sources constitute an example of fixed-size tree sources which are worst-case fringe-dominated:

Lemma 8.33. *Let ℓ be a weight-balanced fixed-size binary tree source. Then*

$$y_{\kappa(n)}(t) = \mathcal{O}\left(\frac{n}{\kappa(n)}\right)$$

for every tree $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$ and function κ, that is, ℓ is worst-case κ-fringe dominated.

Proof. The statement of the lemma follows from known results shown in [42] (see also [S3, Lemma 3]). Let $0 < \beta \leq 1$. In [42], β-balanced binary trees are introduced: A node v of a binary tree $t \in \mathcal{B}^\diamond$ is called β-balanced, if it satisfies $|t_l[v]| + 1 \geq \beta(|t_r[v]| + 1)$ and $|t_r[v]| + 1 \geq \beta(|t_\ell[v]| + 1)$ (note that in [42], the authors count leaves of full binary trees, such that there is an off-by-one in the definition of β-balanced nodes). A binary tree is called β-balanced, if for all internal nodes u, v of t such that u is the parent node of v, we have that u is β-balanced or v is β-balanced. In the proof of [42, Lemma 10], it is shown in the context of DAG-compression of trees that for every β-balanced tree $t \in \mathcal{B}_n^\diamond$, we have $y_k(t) \leq 4\alpha n/k$ for every constant $k \in \mathbb{N}$, where $\alpha = 1 + \log_{1+\beta}(\beta^{-1})$. Now let ℓ be a weight-balanced fixed-size tree source and let $t \in \mathcal{B}^\diamond$ be a binary tree with $P_\ell(t) > 0$. It remains to show that t is β-balanced for some constant β. Let v be a node of t. As $P_\ell(t) > 0$, we find that $\ell(|t_l[v]|, |t_r[v]|) > 0$, and thus, there is a constant γ, such that $\gamma n \leq |t_\ell[v]| + 1, |t_r[v]| + 1 \leq (1-\gamma)n$. In particular, we find that $|t_l[v]| + 1 \geq \gamma(|t_r[v]| + 1)$ and $|t_r[v]| + 1 \geq \gamma(|t_\ell[v]| + 1)$. Thus, t is β-balanced with $\beta = \gamma$. □

Finally, we will present a class of fixed-height binary tree sources that generalizes AVL-trees and is worst-case κ-fringe dominated (and thus amenable to compression using our techniques).

Definition 8.34 (ζ-height-balanced fixed-height sources)**.** A fixed-height tree source ℓ is called *ζ-height-balanced*, if there is an increasing function $\zeta \colon \mathbb{N} \to \mathbb{N}_0$, such that for all $(i, j) \in \mathbb{N}_0 \times \mathbb{N}_0$ with $\ell(i, j) > 0$ and $\max(i, j) = k - 1$ we have $|i - j| \leq \zeta(k)$.

For ζ-height-balanced tree sources, we obtain the following lemma.

Lemma 8.35. *Let ℓ be a ζ-height-balanced fixed-height tree source, then*

$$y_{\kappa(n)}(t) \leq \mathcal{O}\left(\frac{\zeta(n)n\log\kappa(n)}{\kappa(n)}\right)$$

for every tree $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$ and function κ.

In particular, under the assumption that $\kappa(n) = \Theta(\log n)$, ℓ is worst-case fringe-dominated if $\zeta(k) \leq o(\log k/(\log\log k)^2)$. The class of ζ-height-balanced fixed-height tree sources generalizes depth-balanced tree sources introduced in [S3]. The fixed-height binary tree source from Example 8.21 is an example of a 1-height-balanced fixed-height tree source.

Lemma 8.35 follows from combining, respectively, generalizing results from [S3, Lemma 7] and [54, Lemma 2], the latter presented in the context of top-tree compression; a self-contained proof in our notation can be found in [S7]. The lemma from [54, Lemma 2] reads as follows in our notation:

Lemma 8.36 ([54, Lemma 2], [S7, Lemma E.19])**.** *Let $t \in \mathcal{B}^\diamond$ be a binary tree and let $k \in \mathbb{N}$. If there is a constant $c > 1$, such that $\hbar(t[v]) \leq \log_c(|t[v]|+1)$ for every node v of t, then the number $y_k(t)$ of nodes v with $|t[v]| \geq k$ in t satisfies*

$$y_k(t) \leq \frac{4|t|(\log k + 2)}{k\log c} + \frac{2|t|}{k}.$$

With Lemma 8.36, we are able to prove Lemma 8.35.

Proof of Lemma 8.35. Let $\beta \in \mathbb{N}$. We call a binary tree t *β-height-balanced*, if for every node v of t, we have $|\hbar(t_l[v]) - \hbar(t_r[v])| \leq \beta$. This property of trees is called β-depth-balanced trees in [S3]. Note that every subtree of a β-height-balanced tree is β-height-balanced as well. In [S3, Lemma 7], it is shown that for every β-height-balanced tree t, we have $|t| + 1 \geq c^{h(t)}$ with $c = 1 + 1/(1+\beta)$ (note that in [S3], the authors consider full binary trees and measure size as the number of leaves, such that there is an off-by-one in the meaning of $|t|$). Thus, Lemma 8.36 applies to β-height-balanced trees. Now let ℓ be a fixed-height tree source, and let $\zeta : \mathbb{N} \to \mathbb{N}_0$ be a monotonically increasing function, such that for all $(i,j) \in \mathbb{N}_0 \times \mathbb{N}_0$ with $\ell(i,j) > 0$ and $\max(i,j) = k-1$, we have $|i-j| \leq \zeta(k)$. Moreover, let $t \in \mathcal{B}_n^\diamond$ be a binary tree of size n with $P_\ell(t) > 0$. Then $|\hbar(t_l[v]) - \hbar(t_r[v])| \leq \zeta(\hbar(t[v]))$ for every node v of t. In particular, as ζ is increasing, we find that t is β-height balanced with $\beta = \zeta(\hbar(t))$ and as $\hbar(t) \leq |t| = n$, t is $\zeta(n)$-height-balanced. By Lemma 8.36, we thus find that

$$y_{\kappa(n)}(t) \leq \frac{4n(\log\kappa(n)+2)}{\kappa(n)\log c} + \frac{2n}{\kappa(n)},$$

with $c = 1 + 1/(1+\zeta(n))$. By the mean-value theorem, we find

$$\log\left(1 + \frac{1}{1+\zeta(n)}\right) = \log(2+\zeta(n)) - \log(1+\zeta(n)) \geq \frac{1}{(2+\zeta(n))\ln(2)}.$$

Thus

$$y_{\kappa(n)}(t) \leq \frac{4\ln(2)(2+\zeta(n))n(\log\kappa(n)+2)}{\kappa(n)} + \frac{2n}{\kappa(n)} = \mathcal{O}\left(\frac{\zeta(n)n\log\kappa(n)}{\kappa(n)}\right).$$

This proves the lemma. □

8.4.5 Universality with respect to fixed-size binary tree sources and fixed-height binary tree sources

In order to show universality of the hypersuccinct code from Section 8.2 with respect to fixed-size and fixed-height binary tree sources, we proceed in a similar way as in the case of node-type sources. First, we derive a source-specific encoding (called depth-first order arithmetic code) with respect to the fixed-size or fixed-height source, against which we will then compare our hypersuccinct code.

As in the case of node-type sources, we find that the formulas for $P_\ell(t)$, (8.7) and (8.8), immediately suggest a route for an (essentially) optimal *source-specific* encoding of any binary tree t that, given a fixed-size or fixed-height source ℓ, spends $\log(1/P_\ell(t))$ (plus lower-order terms) many bits in order to encode a binary tree $t \in \mathcal{B}^\diamond$ with $P_\ell(t) > 0$. For a given fixed-size source, such an encoding may spend $-\log\big(\ell(|t_l[v]|, |t_r[v]|)\big)$ many bits per node v, while for a fixed-height source, it may spend $-\log\big(\ell(\hbar(t_l[v]), \hbar(t_r[v]))\big)$ many bits per node v.

Assuming that ℓ need not be stored as part of the encoding, and assuming that we have already stored $|t[v]|$, if ℓ corresponds to a fixed-size source, respectively, $\hbar(t[v])$, if ℓ corresponds to a fixed-height source, we can use again *arithmetic coding* to store $|t_l[v]|$ (from which we will then be able to determine $|t_r[v]|$), if ℓ corresponds to a fixed-size source, respectively, $\hbar(t_l[v])$ and $\hbar(t_r[v])$, if ℓ corresponds to a fixed-height source. First, let us assume that ℓ corresponds to a fixed-size binary tree source. A simple (source-dependent) encoding D_ℓ thus stores a tree $t \in \mathcal{B}^\diamond_n$ as follows. We initially encode the size of the tree in Elias gamma code. If the tree consists of n nodes, we store the Elias gamma code of $n+1$, $\gamma(n+1)$, in order to take the case into account that t is the empty binary tree. Additionally, while traversing the tree in depth-first order, we encode $|t_l[v]|$ for each node v, using arithmetic coding. To encode $|t_l[v]|$, we feed the arithmetic coder with the model that the next symbol is a number $i \in \{0, \ldots, |t[v]| - 1\}$ with respective probabilities $\ell(i, |t[v]| - 1 - i)$.

If ℓ corresponds to a fixed-height binary tree source, we proceed similarly. A (source-dependent) encoding D_ℓ with respect to a fixed-height source ℓ stores a tree $t \in \mathcal{B}^\diamond$ of height n by initially encoding $n+1$, in Elias gamma code, followed by an encoding of $(\hbar(t_l[v]), \hbar(t_r[v]))$ for every node v in depth-first order, stored using arithmetic encoding. Note that there are $2\hbar(t[v]) - 1$ many different possibilities for $(\hbar(t_l[v]), \hbar(t_r[v]))$, thus, we can represent a pair $(\hbar(t_l[v]), \hbar(t_r[v]))$ by a number $i \in \{0, 2\hbar(t[v]) - 2\}$.

We refer to this (source-dependent) code D_ℓ as the *depth-first arithmetic code* for the binary tree source ℓ. We can reconstruct the tree t recursively from

its code $D_\ell(t)$: Since we always know the subtree size, respectively, subtree height, we know how many and what size the bins for the next left subtree size, respectively, pair of subtree heights, use in the arithmetic code.

In total, D_ℓ needs at most $\log(1/P_\ell(t)) + 2$ bits to store t when we know $|t|$, respectively $\hbar(t)$ (depending on the type of tree source). With $\hbar(t) \leq |t|$, and as the Elias-gamma code satisfies $|\gamma(n)| \leq 2\lfloor \log(n) \rfloor + 1$, we find that the total encoding length is upper-bounded by

$$|D_\ell(t)| \leq \log(1/P_\ell(t)) + 2\lfloor \log(|t|+1) \rfloor + 3. \tag{8.9}$$

Universality for monotonic fixed-size and fixed-height sources. We first show universality of our hypersuccinct code with respect to *monotonic* fixed-size and fixed-height sources, as defined in Definition 8.22. We start with the following lemma.

Lemma 8.37. *Let ℓ be a fixed-size or fixed-height tree source and let $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$. If ℓ is monotonic, then*

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right),$$

where Ψ is a Huffman code for the sequence of micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ obtained from the tree covering scheme.

Proof. Let us denote by $D_\ell \colon \mathcal{B}^\diamond \to \{0,1\}^*$ the depth-first arithmetic code as introduced in the beginning of this section. In particular, by Lemma 8.23, we find that $P_\ell(\mathfrak{t}_i) \geq P_\ell(t)$ for all micro trees $\mathfrak{t}_i$ of t, and thus, $D_\ell(\mathfrak{t}_i)$ is well-defined for every micro tree $\mathfrak{t}_i$. Restricting D_ℓ to $\Sigma_{s_{\max}}$ yields a prefix-free code for $\Sigma_{s_{\max}}$, so we know by the optimality of Huffman codes that

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \sum_{i=1}^{m} |D_\ell(\mathfrak{t}_i)|.$$

By our estimate (8.9) for $|D_\ell|$, we find that

$$\begin{aligned}\sum_{i=1}^{m} |D_\ell(\mathfrak{t}_i)| &\leq \sum_{i=1}^{m} \left(\log\left(\frac{1}{P_\ell(\mathfrak{t}_i)}\right) + 3 + 2\lfloor \log(|\mathfrak{t}_i|+1) \rfloor\right) \\ &\leq \sum_{i=1}^{m} \log\left(\frac{1}{P_\ell(\mathfrak{t}_i)}\right) + \mathcal{O}(m \log s_{\max}).\end{aligned}$$

Note that the subtrees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ form a partition of t in the sense that every node of t belongs to exactly one subtree $\mathfrak{t}_i$. Thus, and as ℓ corresponds to a monotonic fixed-size or fixed-height source, we find by Lemma 8.23

$$\sum_{i=1}^{m} \log\left(\frac{1}{P_\ell(\mathfrak{t}_i)}\right) + \mathcal{O}(m \log s_{\max}) \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}(m \log s_{\max}).$$

Altogether, with $m = \Theta(n/\log n)$ and $s_{\max} = \Theta(\log n)$ (see Section 8.2), the statement follows. □

From Lemma 8.37 and Lemma 8.5, we obtain the following result for monotonic tree sources.

Theorem 8.38. *Let ℓ be a monotonic fixed-size or fixed-height tree source. Then the hypersuccinct code $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)| \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right)$$

for every $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$.

Using the terminology from [105], [S3], Theorem 8.38 states that for every monotonic fixed-size or fixed-height binary tree source, the worst-case redundancy (8.1) of the hypersuccinct code H converges to zero (as $n \to \infty$); an analogous result holds with respect to the tree encoder E_{tslp} based on TSLPs from [S3]. The tree sources from Example 8.16, Example 8.17, and Example 8.19 are monotonic fixed-size binary tree sources. Thus, together with Theorem 8.38, we obtain the following corollary.

Corollary 8.39. *The hypersuccinct code $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ satisfies the following bounds.*

(i) With respect to the binary search tree model (see Example 8.16), we find that a tree $t \in \mathcal{B}^\diamond$ of size n is encoded using

$$|\mathsf{H}(t)| \leq \log(1/P_{\ell_{bst}}(t)) + \mathcal{O}(n \log\log n/\log n)$$

many bits. In particular, we need on average

$$\sum_{t \in \mathcal{B}_n^\diamond} P_{\ell_{bst}}(t)|\mathsf{H}(t)| \approx 1.736n + \mathcal{O}(n \log\log n/\log n)$$

many bits (see [66]) in order to encode a binary search tree of size n.

(ii) Almost-path binary trees (for arbitrary $K \geq 0$) from Example 8.19 are encoded using $|\mathsf{H}(t)| \leq \log\big(1/P_{\ell_{path}}(t)\big) + \mathcal{O}(n \log\log n/\log n)$ many bits.

As the uniform probability distribution on the set $\mathcal{B}_n^\diamond$ of binary trees of size n can be modeled as a monotonic fixed-size binary tree source (see Example 8.17), we find moreover that Corollary 8.15, part (i) follows from Theorem 8.38.

Universality for fringe-dominated fixed-size and fixed-height sources. Recall that the hypersuccinct encoding from Section 8.2 decomposes t into micro trees $\mathsf{t}_1, \ldots, \mathsf{t}_m$ using Lemma 8.2 and uses a Huffman code Ψ for $\mathsf{t}_1, \ldots, \mathsf{t}_m$. Some of these micro trees might be *"fringe"*, that is, correspond to fringe subtrees of t and leaves in the top tier tree Υ, but many will be *internal* micro trees, that is, have child micro trees in the top tier tree Υ. That means, micro-tree-local

subtree sizes, resp. heights, and global subtree sizes, resp., heights, differ for nodes that are ancestors of the portal to the child micro tree – and only for those nodes do they differ. This will be the crucial observation in order to show that our hypersuccinct code is universal with respect to fringe-dominated sources.

Formally, let v be a node of t. If v is contained in a fringe micro tree $\mathfrak{t}_i$, respectively, in a non-fringe micro tree $\mathfrak{t}_i$ but not an ancestor of a portal node, then $\mathfrak{t}_i[v] = t[v]$, and thus $\ell(|\mathfrak{t}_{il}[v]|, |\mathfrak{t}_{ir}[v]|) = \ell(|t_l[v]|, |t_r[v]|)$, respectively, $\ell(\hbar(\mathfrak{t}_{il}[v]), \hbar(\mathfrak{t}_{ir}[v])) = \ell(\hbar(t_l[v]), \hbar(t_r[v]))$.

On the other hand, if v is an ancestor of a portal node in a non-fringe subtree $\mathfrak{t}_i$, then $\mathfrak{t}_i[v] \neq t[v]$. In order to take this observation into consideration, we make the following definitions. Let $\mathfrak{t}_i$ be an internal (non-fringe) micro tree. By $\mathfrak{b}(\mathfrak{t}_i)$, we denote the subtree of $\mathfrak{t}_i$ induced by the set of nodes that are ancestors of $\mathfrak{t}_i$'s child micro trees (ancestors of the portals); we refer to $\mathfrak{b}(\mathfrak{t}_i)$ as the *bough* of $\mathfrak{t}_i$. The boughs of a micro tree are the paths from the portals to the micro tree root.

In particular, if v denotes a node of t contained in a subtree $\mathfrak{t}_i$, then $t[v] \neq \mathfrak{t}_i[v]$ if and only if $\mathfrak{t}_i$ is not fringe and v is contained in $\mathfrak{b}(\mathfrak{t}_i)$. Hanging off the boughs of $\mathfrak{t}_i$ are (fringe) subtrees $\mathfrak{f}_{i,1}, \ldots, \mathfrak{f}_{i,|\mathfrak{b}(\mathfrak{t}_i)|+1}$, listed in depth-first order of the bough nulls these subtrees are attached to. In particular, some of these subtrees might be the empty tree. Recall that the portal nodes themselves are not part of $\mathfrak{t}_i$ and hence not part of $\mathfrak{b}(\mathfrak{t}_i)$. We obtain the following lemma.

Lemma 8.40. *Let ℓ be a fixed-size, respectively, fixed-height binary tree source. Furthermore, let $\mathcal{I}_0 = \{i \in \{1, \ldots, m\} \mid \mathfrak{t}_i$ is a fringe micro tree in $t\}$ and let $\mathcal{I}_1 = \{1, \ldots, m\} \setminus \mathcal{I}_0$. Then*

$$\sum_{i \in \mathcal{I}_0} \log\Big(\frac{1}{P_\ell(\mathfrak{t}_i)}\Big) + \sum_{i \in \mathcal{I}_1} \sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1} \log\Big(\frac{1}{P_\ell(\mathfrak{f}_{i,j})}\Big) \leq \log\Big(\frac{1}{P_\ell(t)}\Big).$$

Proof. The statement follows immediately from the facts that (i) all the subtrees $\mathfrak{t}_i$ for $i \in \mathcal{I}_0$ and $\mathfrak{f}_{i,j}$ for $i \in \mathcal{I}_1$ and $j \in \{1, \ldots, |\mathfrak{b}(\mathfrak{t}_i)| + 1\}$ are fringe subtrees of t, and (ii) every node v of t occurs in at most one of these fringe subtrees. Assume that ℓ corresponds to a fixed-size tree source, then we find

$$\begin{aligned}
&\sum_{i \in \mathcal{I}_0} \log\Big(\frac{1}{P_\ell(\mathfrak{t}_i)}\Big) + \sum_{i \in \mathcal{I}_1} \sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1} \log\Big(\frac{1}{P_\ell(\mathfrak{f}_{i,j})}\Big) \\
&= -\sum_{i \in \mathcal{I}_0} \sum_{v \in V(\mathfrak{t}_i)} \log(\ell(|\mathfrak{t}_{il}[v]|, |\mathfrak{t}_{ir}[v]|)) - \sum_{i \in \mathcal{I}_1} \sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1} \sum_{v \in V(\mathfrak{f}_{i,j})} \log(\ell(|\mathfrak{f}_{i,j_l}[v]|, |\mathfrak{f}_{i,j_r}[v]|)) \\
&\overset{(i)}{=} -\sum_{i \in \mathcal{I}_0} \sum_{v \in V(\mathfrak{t}_i)} \log(\ell(|t_l[v]|, |t_r[v]|)) - \sum_{i \in \mathcal{I}_1} \sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1} \sum_{v \in V(\mathfrak{f}_{i,j})} \log(\ell(|t_l[v]|, |t_r[v]|)) \\
&\overset{(ii)}{\leq} -\sum_{v \in V(t)} \log(\ell(|t_l[v]|, |t_r[v]|)) = \log\Big(\frac{1}{P_\ell(t)}\Big).
\end{aligned}$$

The proof for fixed-height sources is similar. □

With these considerations, we are now able to derive the following lemma.

Lemma 8.41. *Let ℓ be a fixed-size, respectively, fixed-height binary tree source and let $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$. Then*

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}\left(y_\kappa(t) \log\log n\right),$$

where Ψ is a Huffman code for the sequence of micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ from our tree covering scheme and $\kappa = \kappa(n) = \Theta(\log n)$ is the parameter of the tree covering scheme.

Proof. We construct a new encoding for micro trees $\bar{D}_\ell$, against which we can compare the hypersuccinct code, as follows.

$$\bar{D}_\ell(\mathfrak{t}_i) = \begin{cases} 0 \cdot D_\ell(\mathfrak{t}_i), & \text{if } \mathfrak{t}_i \text{ is fringe;} \\ 1 \cdot \gamma(|\mathfrak{b}(\mathfrak{t}_i)|) \cdot BP(\mathfrak{b}(\mathfrak{t}_i)) \cdot D_\ell(\mathfrak{f}_{i,1}) \cdots D_\ell(\mathfrak{f}_{i,|\mathfrak{b}(\mathfrak{t}_i)|+1}), & \text{otherwise,} \end{cases}$$

where $D_\ell \colon \mathcal{B}^\diamond \to \{0,1\}^*$ is the depth-first order arithmetic code as introduced in the beginning of Section 8.4.5. Note that $\bar{D}_\ell$ is well-defined, as the encoding D_ℓ is only applied to fringe subtrees $\mathfrak{t}_i$ and $\mathfrak{f}_{i,j}$ of t, for which $P_\ell(\mathfrak{t}_i), P_\ell(\mathfrak{f}_{i,j}) > 0$ follows from $P_\ell(t) > 0$.

Moreover, note that formally, $\bar{D}_\ell$ is *not* a prefix-free code over $\Sigma_{s_{\max}}$, as there can be micro tree shapes that are assigned *several* codewords by $\bar{D}_\ell$, depending on which nodes are portals to other micro trees (if any). But $\bar{D}_\ell$ is uniquely decodable to local shapes of micro trees, and can thus be seen as a *generalized prefix-free code*, and a Huffman code minimizes the encoding length over the larger class of generalized prefix-free codes. In particular, the Huffman code Ψ for micro trees used in the hypersuccinct code achieves no worse encoding length than $\bar{D}_\ell$. We thus have

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \sum_{i=1}^{m} |\bar{D}_\ell(\mathfrak{t}_i)|.$$

We again set $\mathcal{I}_0 = \{i \in \{1, \ldots, m\} \mid \mathfrak{t}_i \text{ is a fringe micro tree in } t\}$, and define $\mathcal{I}_1 = \{1, \ldots, m\} \setminus \mathcal{I}_0$. We have

$$\begin{aligned} \sum_{i=1}^{m} |\bar{D}_\ell(\mathfrak{t}_i)| &= \sum_{i \in \mathcal{I}_0} |\bar{D}_\ell(\mathfrak{t}_i)| + \sum_{i \in \mathcal{I}_1} |\bar{D}_\ell(\mathfrak{t}_i)| \\ &\leq \sum_{i \in \mathcal{I}_0} (1 + |D_\ell(\mathfrak{t}_i)|) + \sum_{i \in \mathcal{I}_1} \Big(2 + 2\log(|\mathfrak{b}(\mathfrak{t}_i)|) + 2|\mathfrak{b}(\mathfrak{t}_i)| + \sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1} |D_\ell(\mathfrak{f}_{i,j})|\Big). \end{aligned}$$

With the estimate (8.9), we obtain for the first sum

$$\sum_{i \in \mathcal{I}_0} (1 + |D_\ell(\mathfrak{t}_i)|) \leq \sum_{i \in \mathcal{I}_0} \log\Big(\frac{1}{P_\ell(\mathfrak{t}_i)}\Big) + \mathcal{O}(m \log s_{\max}).$$

Moreover, also with the estimate (8.9), we find for the second sum

$$\begin{aligned}
&\sum_{i\in\mathcal{I}_1}\Big(2+2\log(|\mathfrak{b}(\mathfrak{t}_i)|)+2|\mathfrak{b}(\mathfrak{t}_i)|+\sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1}|D_\ell(\mathfrak{f}_{i,j})|\Big)\\
&\le\sum_{i\in\mathcal{I}_1}\sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1}\Big(3+\log\Big(\frac{1}{P_\ell(\mathfrak{f}_{i,j})}\Big)+2\log(|\mathfrak{f}_{i,j}|+1)\Big)+\mathcal{O}\Big(\sum_{i\in\mathcal{I}_1}|\mathfrak{b}(\mathfrak{t}_i)|\Big)\\
&\le\sum_{i\in\mathcal{I}_1}\sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1}\log\Big(\frac{1}{P_\ell(\mathfrak{f}_{i,j})}\Big)+\mathcal{O}\Big(\sum_{i\in\mathcal{I}_1}|\mathfrak{b}(\mathfrak{t}_i)|\log s_{\max}\Big).
\end{aligned}$$

By Lemma 8.40, we have

$$\sum_{i\in\mathcal{I}_0}\log\Big(\frac{1}{P_\ell(\mathfrak{t}_i)}\Big)+\sum_{i\in\mathcal{I}_1}\sum_{j=1}^{|\mathfrak{b}(\mathfrak{t}_i)|+1}\log\Big(\frac{1}{P_\ell(\mathfrak{f}_{i,j})}\Big)\le\log\Big(\frac{1}{P_\ell}\Big).$$

It remains to upper-bound the error terms: Lemma 8.3 implies that any node v in the bough of a micro tree satisfies $|t[v]|\ge\kappa$. Thus, the total number of nodes of t which belong to a bough of t is therefore upper-bounded by $y_\kappa(t)$. Altogether, we thus obtain

$$\sum_{i=1}^{m}|\Psi(\mathfrak{t}_i)|\le\log\left(\frac{1}{P_\ell(t)}\right)+\mathcal{O}\left(m\log s_{\max}\right)+\mathcal{O}\big(y_\kappa(t)\log s_{\max}\big).$$

By a pigeon-hole argument, we find $y_\kappa(t)=\Omega(n/\kappa)$. As $s_{\max}=\Theta(\log n)$ and $m=\Theta(n/\log n)$, the statement follows. □

For average-case fringe-dominated fixed-size binary tree sources (defined in Definition 8.24), we obtain the following result from Lemma 8.41 and Lemma 8.5.

Theorem 8.42. *Let ℓ be an average-case κ-fringe-dominated fixed-size binary tree source, where $\kappa(n)=\lceil\frac{1}{8}\log(n)\rceil$. Then the hypersuccinct code H satisfies*

$$\sum_{t\in\mathcal{B}_n^\diamond}P_\ell(t)|\mathsf{H}(t)|\le\sum_{t\in\mathcal{B}_n^\diamond}P_\ell(t)\log\left(\frac{1}{P_\ell(t)}\right)+o(n).$$

For worst-case fringe-dominated fixed-size, respectively, fixed-height binary tree sources (defined in Definition 8.25), we get the following result from Lemma 8.41 and Lemma 8.5.

Theorem 8.43. *Let ℓ be a worst-case κ-fringe-dominated fixed-size or fixed-height binary tree source, where $\kappa(n)=\lceil\frac{1}{8}\log(n)\rceil$. Then the hypersuccinct code $\mathsf{H}\colon\mathcal{B}^\diamond\to\{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)|\le\log\left(\frac{1}{P_\ell(t)}\right)+o(n)$$

for every binary tree $t\in\mathcal{B}_n^\diamond$ with $P_\ell(t)>0$.

That is, in the terminology from [105] and [S3], Theorem 8.42 (respectively, Theorem 8.43) states that for every κ-average-case fringe-dominated fixed-size source (respectively, every κ-worst-case fringe-dominated fixed-size or fixed-height source), the average-case redundancy (8.2) (respectively, worst-case redundancy (8.1)) of H converges to zero. Similar results were shown for the tree encoder E_{dag} based on DAGs in [105] and [S3].

In Section 8.4.4, we have presented several general classes of fixed-size and fixed-height tree sources, which are average-case or worst-case fringe-dominated. For these classes, we now obtain the following universality results of our hypersuccinct encoding. First, we obtain the following corollary for ψ-upper-bounded fixed-size sources (Definition 8.26), φ-weakly-balanced fixed-size sources (Definition 8.27), and ς-strongly-balanced fixed-size sources (Definition 8.28) using Lemma 8.29, Lemma 8.30 and Lemma 8.31.

Corollary 8.44. *Let ℓ be a fixed-size binary tree source. If ℓ is ψ-upper-bounded, or φ-weakly-balanced, or ς-strongly-balanced, then the hypersuccinct code $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ satisfies*

$$\sum_{t\in\mathcal{B}_n^\diamond} P_\ell(t)|\mathsf{H}(t)| \leq \sum_{t\in\mathcal{B}_n^\diamond} P_\ell(t)\log\left(\frac{1}{P_\ell(t)}\right) + o(n).$$

More precisely, the redundancy term $o(n)$ evalutates to $\mathcal{O}(n\psi(\log n)\log\log n)$, if ℓ is ψ-upper-bounded, or $\mathcal{O}(n\log\log n/(\varphi(n)\log n))$, if ℓ is φ-weakly-balanced, or $\mathcal{O}(n\log\log n/(\varsigma(\log n)\log n))$, if ℓ is ς-strongly-balanced. Moreover, with Lemma 8.33, we find for weight-balanced fixed-size binary tree sources (defined in Definition 8.32):

Corollary 8.45. *Let ℓ be a weight-balanced fixed-size binary tree source. Then the hypersuccinct code $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)| \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}\left(\frac{n\log\log n}{\log n}\right)$$

for every binary tree $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$.

Finally, with Lemma 8.35, we obtain for ζ-height-balanced fixed-height binary tree sources (defined in Definition 8.34):

Corollary 8.46. *Let ℓ be a ζ-height-balanced fixed-height binary tree source. Then the hypersuccinct code $\mathsf{H}\colon \mathcal{B}^\diamond \to \{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)| \leq \log\left(\frac{1}{P_\ell(t)}\right) + \mathcal{O}\left(\frac{\zeta(n)n\log\log n}{\log n}\right)$$

for every binary tree $t \in \mathcal{B}_n^\diamond$ with $P_\ell(t) > 0$.

Additionally, as the fixed-size and fixed-height tree sources from Example 8.18, Example 8.20 and Example 8.21 are (average-case or worst-case) fringe dominated, we moreover obtain the following corollary from Theorem 8.42 and Theorem 8.43 for these particular sources.

Corollary 8.47. *The hypersuccinct code* H *satisfies the following upper bounds.*

(i) A binary tree of size n *randomly generated by the* ***binomial random tree model*** $\ell_{bin,p}$ *from Example 8.18 or the* ***random fringe-balanced binary search tree model*** ℓ_{bal} *from Example 8.20 is average-case optimally encoded. That is, for* $\ell \in \{\ell_{bin,p}, \ell_{bal}\}$*, we have*

$$\sum_{t\in\mathcal{B}_n^\diamond} P_\ell(t)|\mathsf{H}(t)| \leq \sum_{t\in\mathcal{B}_n^\diamond} P_\ell(t)\log\left(\frac{1}{P_\ell(t)}\right) + o(n).$$

(iii) An ***AVL tree*** t *of size* n *and height* h*, drawn uniformly at random from the set* $\mathcal{A}^h$ *of all AVL trees of height* h*, is optimally compressed using* $|\mathsf{H}(t)| \leq \log(|\mathcal{A}^h|) + o(n)$ *many bits (see Example 8.21).*

We remark that the average-case result from Corollary 8.39, part (i), also follows from Theorem 8.42.

8.5 Conclusion and open problems

Further universality results with respect to hypersuccinct trees as well as the data structure aspects can be found in [S7]. In particular, in [S7], it is shown that the hypersuccinct tree encoding is (worst-case) universal with respect to a class of sources called *tame uniform-subclass sources*, which allow to model uniform distributions on families of binary trees. For example, the uniform distribution on the set of AVL trees of *size* n (see Example 8.21) and the uniform distribution on the set of (left-leaning) red-black trees of size n can be modeled using uniform subclass-sources. The proof of universality of the hypersuccinct encoding H with respect to these sources is conceptionally quite similar to the proofs of Theorem 8.42 and Theorem 8.43.

In contrast to the universality results for label-shape processes (Theorem 6.20) and for node-type sources (Theorem 8.14), the universality results for fixed-size binary tree sources and fixed-height binary tree sources (Theorem 8.38, Theorem 8.42 and Theorem 8.43) do not seem to imply an entropy bound. For this one would first have to come up with a suitable entropy notion that is related to these tree sources. Moreover, in [105], it is shown that leaf-centric and depth-centric binary tree sources as general classes of sources do not admit a universal code; thus, making suitable restrictions is necessary.

In Chapter 6, we have introduced the notion of label-shape entropy for the family of *full* binary trees (Definition 6.14). Recall that label-shape entropy is also reasonable for unlabeled binary trees. It seems quite natural to transfer the concept of label-shape entropy and label-shape processes from $\mathcal{B}$ to $\mathcal{B}^\diamond$; this would be conceptionally quite similar to node-type sources from Section 8.3, such that an entropy bound for H in terms of label-shape entropy should be provable.

Another natural topic for future research is to generalize the results of this chapter from unlabeled binary trees to labeled binary trees. A generalization of hypersuccinct trees to the family of plane trees is presented in the next chapter.

Chapter 9

Hypersuccinct plane trees

9.1 Introduction

The goal of this chapter is to present a variant of the hypersuccinct encoding introduced in the last chapter for the family of unlabeled plane trees $\mathcal{T}$, which we call *hypersuccinct plane trees*. As in the case of hypersuccinct binary trees, this tree encoding is again based on the tree decomposition algorithm by Farzan and Munro from [30] and can be turned into a compressed data structure for unlabeled plane trees that supports answering many navigational queries on the compressed representation in constant time on the word-RAM.

What is more, hypersuccinct plane trees can be shown to be universal with respect to several classes of sources for plane trees. As discussed in Section 8.1, universal tree source coding has so far only been considered with respect to the family of full binary trees in [105] and [S3] (see also [66]); we are not aware of similar works specifically focusing on plane trees. However, entropy bounds for compressed representations of unlabeled plane trees are known, as the entropy bound in terms of the *degree entropy* (Definition 7.2) from [60] and the entropy bound in terms of the *label-shape entropy* (Definition 6.22) presented in Theorem 6.23.

Hypersuccinct plane trees achieve an entropy bound both in terms of the degree entropy as well as in terms of the label-shape entropy. In order to derive an entropy bound in terms of the degree entropy, we show that hypersuccinct plane trees are (worst-case) universal with respect to the class of *Galton–Watson processes*, which were considered in Chapter 4 (see Definition 4.7) of this work. In fact, it will turn out that Galton–Watson processes are related to the degree entropy in the same way as label-shape processes relate to label-shape entropy and node-type sources relate to node-type entropy (see Theorem 6.18 and Theorem 8.7). Furthermore, hypersuccinct plane trees can be shown to be universal with respect to another class of sources: In [S7], fixed-size binary tree sources (resp., leaf-centric binary tree sources) were generalized to *fixed-size ordinal tree sources*, which generate (unlabeled) plane trees. Hypersuccinct plane trees are shown to be universal with respect to particular subclasses of fixed-size

ordinal tree sources in [S7], similar to the classes of monotonic and fringe-dominated fixed-size binary tree sources from Section 8.4.3 and Section 8.4.4.

As in the previous chapter, we focus on a selection of universality results and entropy bounds with respect to hypersuccinct plane trees here; for data structure aspects and a detailed overview over various compressed tree data structures presented in the literature, as well as further universality results, we refer again to [S7]. In Section 9.2, we introduce our hypersuccinct encoding of plane trees. In Section 9.3 and Section 9.4, we show that hypersuccinct plane trees are universal with respect to the class of Galton–Watson processes and achieve entropy bounds in terms of the degree entropy and the label-shape entropy. The concepts and techniques considered in this chapter are conceptionally quite similar to the concepts and techniques from the previous chapter. The results of this chapter are published in [S7].

9.2 Hypersuccinct encoding of plane trees

As in the previous chapter, we make use of *Huffman coding* [20, 55], *arithmetic coding* [20, 104] and the *Elias gamma code* [29], where we denote the Elias gamma encoding of $n \in \mathbb{N}$ with $\gamma(n)$. In this chapter, we focus on the family of (unlabeled) plane trees $\mathcal{T}$. A tree $t \in \mathcal{T}$ of size n can be encoded using $2n$ bits via a variant of the balanced parenthesis encoding for plane trees:

Definition 9.1 (Balanced parenthesis encoding for plane trees)**.** We define the *balanced parenthesis encoding* for plane trees $BP\colon \mathcal{T} \to \{(,)\}^*$ inductively by

$$BP(t) = \begin{cases} \epsilon & \text{if } t \text{ is the empty tree,} \\ (BP(t_1)BP(t_2)\cdots BP(t_r)) & \text{otherwise.} \end{cases}$$

Here, ϵ again denotes the empty string and $t_1, \ldots, t_r$ denote the root branches of the tree t.

The Farzan-Munro tree decomposition algorithm [30] that we have already used in our hypersuccinct encoding for binary trees is originally defined for plane trees. It satisfies the following properties in the setting of plane trees.

Lemma 9.2 (Tree covering, [30, Thm. 1])**.** *For any parameter $\kappa \geq 1$, a plane tree with n nodes can be decomposed, in linear time, into connected subtrees (called* micro trees*) with the following properties:*

- *Micro trees are pairwise disjoint except for (potentially) sharing a common micro tree root.*
- *Each micro tree contains at most 2κ nodes.*
- *The overall number of micro trees is $\Theta(n/\kappa)$.*
- *Apart from edges leaving the micro tree root, at most one other edge leads to a node outside of this micro tree. This edge is called the "external edge" of the micro tree.*

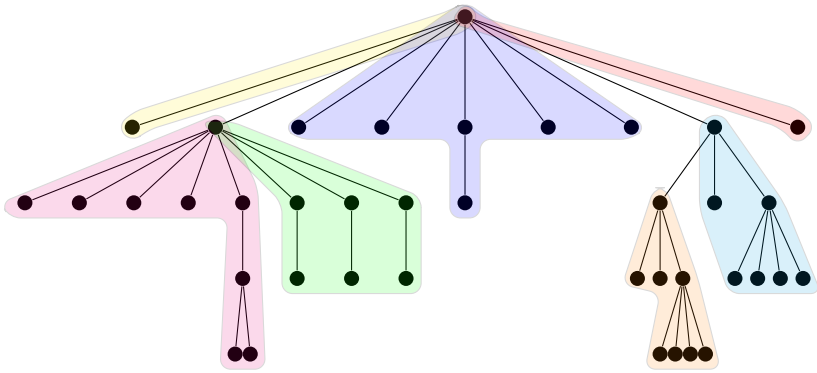


Figure 9.1: A plane tree with $n = 39$ nodes, decomposed into six micro trees (highlighted in different colours) by the tree decomposition algorithm from [30] with $\kappa = 9$.

By inspection of the proof in [30], we can say a bit more. If v is a node in the tree and is also the root of several micro trees of the decomposition, then the way that v's children (in the entire tree) are divided among the micro trees is into *consecutive* blocks (of micro trees). Each micro tree contains at most two of these blocks. In binary trees, a micro tree is always an entire fringe subtree except for at most two entire fringe subtrees, which are removed from it. In plane trees, the possibility of large node degrees makes such a decomposition impossible. Here an arbitrary number of children (and their subtrees) can be missing in a micro tree root, and a single node in the original tree can be the (shared) root of many micro trees.

Hypersuccinct code. We next describe the hypersuccinct encoding for plane trees based on the Farzan-Munro algorithm. We fix the parameter κ, so that the maximal micro tree size is $s_{\max} = \lceil \frac{1}{4} \log n \rceil$, that is, we set $\kappa = \lceil \frac{1}{8} \log n \rceil$. The encoding of the plane tree $t \in \mathcal{T}_n$ is then obtained as follows. Decompose the tree into micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ where $m = \Theta(n/\kappa) = \Theta(n/\log n)$. Recall that each micro tree $\mathfrak{t}_i$ can have the following connections to other micro trees:

- an edge to one parent micro tree,
- an external edge to one child micro tree, leaving from some node of the micro tree (and inserted at some child rank),
- an arbitrary number of other subtrees of the shared root; these micro trees can contain the shared root or not.

The *top-tier* Υ of the tree is a plane tree, which we obtain by contracting each micro tree into a single node; shared roots are copied to each micro tree. These contracted micro trees are then connected by edges to form a plane tree Υ. We connect contracted micro trees only if there is an edge between some nodes in these micro trees in t. If the parent node of the root node of a micro tree

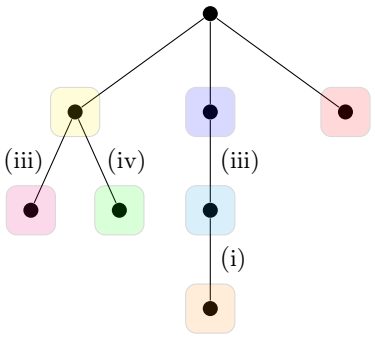


Figure 9.2: The top-tier Υ corresponding to the decomposed tree from Figure 9.1 and the types of edges (i)-(v) connecting parent and child micro trees.

$\mathfrak{t}_i$ is a shared root node (and hence copied into two or more micro trees), the contracted micro tree $\mathfrak{t}_i$'s ancestor in Υ is the parent micro tree immediately to the left (if it exists, otherwise, the parent micro tree immediately to the right) in the left-to-right order of the nodes. Since several micro trees can contain the root of the tree t, we add a dummy root to Υ to turn it into a single tree. The top-tier Υ of the tree from Figure 9.1 is shown in Figure 9.2.

To be able to distinguish the different forms of interactions between the micro trees, additional information for parent-child edges in Υ is stored. By construction, edges between micro trees always lead to the root of the child micro tree, but the other endpoint will have to be encoded. In particular, the following types of edges between a parent micro tree $\mathfrak{p}$ and its child $\mathfrak{c}$ are possible.

(i) *new leftmost root child*
The root of $\mathfrak{c}$ is a child of the root of $\mathfrak{p}$ and comes before all children of $\mathfrak{p}$'s root that lie inside $\mathfrak{p}$ in the left-to-right order of the children. Moreover, there is no other child component $\mathfrak{c}'$ of $\mathfrak{p}$ that shares the root with $\mathfrak{c}$ and comes before $\mathfrak{c}$ in the child order.

(ii) *continued leftmost root child*
The root of $\mathfrak{c}$ is a child of the root of $\mathfrak{p}$ and comes before all children of $\mathfrak{p}$'s root that lie inside $\mathfrak{p}$ in the left-to-right order of the children, but it shares its root with the child component immediately before $\mathfrak{c}$ in the child order.

(iii) *new rightmost root child*
The root of $\mathfrak{c}$ is a child of the root of $\mathfrak{p}$ and $\mathfrak{c}$'s root comes after all root children included in $\mathfrak{p}$. Moreover, there is no other child component $\mathfrak{c}'$ of $\mathfrak{p}$ that shares the root with $\mathfrak{c}$ and comes before $\mathfrak{c}$ in the child order.

(iv) *continued rightmost root child*
The root of $\mathfrak{c}$ is a child of the root of $\mathfrak{p}$ and $\mathfrak{c}$'s root comes after all root children included in $\mathfrak{p}$, but it shares its root with the child component immediately before $\mathfrak{c}$ in the child order.

(v) *external-edge child*
Any other edge. By construction, all external-edge child components of $\mathfrak{p}$ share a common root, so there is no need to distinguish new and continued external edges.

See Figure 9.2 for an example. The top tier is again an plane tree, $\Upsilon \in \mathcal{T}_{m+1}$. For the micro trees, we observe that because of their limited size, there are fewer different possible shapes of plane trees than we have micro trees. The crucial idea of our hypersuccinct encoding is again to treat each shape of a micro tree as a letter in the alphabet $\Sigma_{s_{\max}} \subseteq \bigcup_{s \leq s_{\max}} \mathcal{T}_s$ of micro tree shapes and to compute a Huffman code $\Psi\colon \Sigma_{s_{\max}} \to \{0,1\}^*$ based on the frequency of occurrences of micro tree shapes in the sequence $\mathfrak{t}_1, \ldots, \mathfrak{t}_m \in \Sigma_{s_{\max}}^m$.

For our hypersuccinct code, as in the case of binary trees, we then use a *length-restricted* version $\bar{\Psi}\colon \Sigma_{s_{\max}} \to \{0,1\}^*$ obtained from Ψ using a variant of the simple cutoff technique from Definition 8.4 for plane trees (using the balanced parenthesis encoding for plane trees). Furthermore, for each micro tree, we have to encode the portal for the external edges (if they exist) and the type of its parent edge (i)–(v). For that, we store the micro-tree-local preorder rank of the node and the child rank at which the external edges have to be inserted using $\lceil \log(s_{\max}+1) \rceil$ bits each. We can thus encode an plane tree $t \in \mathcal{T}_n$ as follows.

(a) Store n and m in Elias gamma code,

(b) followed by the balanced-parenthesis bitstring for Υ.

(c) Next comes an encoding for $\bar{\Psi}$; for simplicity, we simply list all possible codewords and their corresponding plane trees by storing the size (in Elias-gamma code) followed by their balanced parenthesis sequence.

(d) Next, we list the Huffman codes $\bar{\Psi}(\mathfrak{t}_i)$ of all micro trees in depth-first order (of the top tier Υ).

(e) Then, we store 2 $\lceil \log(s_{\max}+1) \rceil$-bit integers to encode the portal of each micro tree in depth-first order (of Υ).

(f) Finally, we encode the type of the parent edge using 3 bits of each micro tree, again in depth-first order.

Altogether, this yields our *hypersuccinct code* $\mathsf{H}\colon \mathcal{T} \to \{0,1\}^*$ for plane trees. Decoding is possible by first recovering n, m, and Υ from the balanced parenthesis encoding, then reading the Huffman code. We then replace each node in Υ by its micro tree in a depth-first traversal. Herein, we use the information about edge types in Υ to correctly connect the micro trees. Partitioning children into leftmost and rightmost root children places them in the appropriate order into the list of children of the parent component's root. For type (ii) and (iv) children, we delete the component root and instead add its children to the next type (i) resp. (iii) siblings component's root. Finally, for type (v) children, we use the information about portals to find their place in a node's child list, and

for all but the leftmost of them, also merge their roots with the left sibling component. With respect to the length of the hypersuccinct code, we obtain the following lemma.

Lemma 9.3. *Let $t \in \mathcal{T}_n$ be an plane tree of n nodes, decomposed into micro trees $\mathsf{t}_1, \ldots, \mathsf{t}_m$ by the Farzan-Munro algorithm. Let Ψ be an ordinary Huffman code for the string $\mathsf{t}_1 \ldots \mathsf{t}_m$ (the local shapes of the micro trees). Then, the hypersuccinct code encodes t with a binary codeword of length*

$$|\mathsf{H}(t)| \leq \sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| + \mathcal{O}\Big(\frac{n \log\log n}{\log n}\Big).$$

Proof. It is easy to check that all parts of the hypersuccinct plane tree code except part (d) require $\mathcal{O}(n \log\log n/\log n)$ bits of space. The analysis of the number of bits needed to store parts (a)–(e) is identical to the binary-tree case: Part (a) needs $\mathcal{O}(\log n)$ bits and Part (b) requires $2m + 2 = \Theta(n/\log n)$ bits. For Part (c), observe that

$$|\Sigma_{s_{\max}}| \leq \sum_{s \leq \lceil \log n/4 \rceil} 4^s < \frac{4}{3} \cdot 4^{\log n/4+1} = \frac{16}{3}\sqrt{n}.$$

With the worst-case cutoff technique (adapted to plane trees) from Definition 8.4, we find that $|\bar{\Psi}(\mathsf{t}_i)| \leq 1 + 2s_{\max} \sim \frac{1}{2}\log n$, so we need asymptotically $\mathcal{O}(\sqrt{n})$ entries / codewords in the table, each of size $\mathcal{O}(s_{\max}) = \mathcal{O}(\log n)$, for an overall look-up table size of $\mathcal{O}(\sqrt{n}\log n)$. Furthermore, we find that part (e) uses $m \cdot 2\lceil \log(s_{\max}+1) \rceil = \Theta(\frac{n}{\kappa}\log\kappa) = \Theta(\frac{n\log\log n}{\log n})$ bits of space. Moreover, part (f) uses $3m = \Theta(n/\log n)$ bits. It remains to analyze part (d), which is again similar to the binary-tree case. We note that by applying the worst-case pruning scheme of Definition 8.4, we waste 1 bit per micro tree compared to a pure, non-restricted Huffman code. But the wasted bits amount to $m = O(n/\log n)$ bits in total:

$$\begin{aligned}\sum_{i=1}^{m} |\bar{\Psi}(\mathsf{t}_i)| &= \sum_{i=1}^{m} \min\{|\Psi(\mathsf{t}_i)| + 1, 2|\mathsf{t}_i| + 2\lceil \log|\mathsf{t}_i| + 1 \rceil + 2\} \\ &\leq \sum_{i=1}^{m} (|\Psi(\mathsf{t}_i)| + 1) = \sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| + \mathcal{O}(n/\log n).\end{aligned}$$

This finishes the proof. □

As in the case of hypersuccinct binary trees, the representation of plane trees based on the hypersuccinct code can be turned into a data structure with constant-time queries in the word-RAM model. For the data structure details of hypersuccinct trees, we again refer to [S7]. Specifically, Table 9.1 shows which queries can be supported by hypersuccinct trees in constant time on the word-RAM.

$\texttt{parent}(v)$	the parent of v, same as $\texttt{ancestor}(v, 1)$
$\texttt{deg}(v)$	the number of children of v
$\texttt{child}(v, i)$	the ith child of node v ($i \in \{1, \ldots, \texttt{deg}(v)\}$)
$\texttt{childrank}(v)$	the number of siblings to the left of node v plus 1
$\texttt{depth}(v)$	the depth of v (the number of edges between the root and v)
$\texttt{ancestor}(v, i)$	the ancestor of node v at depth $\texttt{depth}(v) - i$
$\texttt{descendants}(v)$	the number of descendants of v
$\texttt{height}(v)$	the height of the subtree rooted at node v
$\texttt{LCA}(v, u)$	the lowest common ancestor of nodes u and v
$\texttt{leftmostleaf}(v)$	the leftmost leaf descendant of v
$\texttt{rightmostleaf}(v)$	the rightmost leaf descendant of v
$\texttt{levelleftmost}(i)$	the leftmost node on level i
$\texttt{levelrightmost}(i)$	the rightmost node on level i
$\texttt{levelpred}(v)$	the node immediately to the left of v on the same level
$\texttt{levelsucc}(v)$	the node immediately to the right of v on the same level
$\texttt{noderank}_X(v)$	the position of v in the X-order, $X \in \{\texttt{pre}, \texttt{post}, \texttt{in}, \texttt{DFUDS}\}$
$\texttt{nodeselect}_X(i)$	the i^{th} node in the X-order, $X \in \{\texttt{pre}, \texttt{post}, \texttt{in}, \texttt{DFUDS}\}$
$\texttt{leafrank}(v)$	the number of leaves before and including v in preorder
$\texttt{leafselect}(i)$	the ith leaf in preorder

Table 9.1: Navigational operations on succinct plane trees (v denotes a node and i an integer).

9.3 Degree entropy and Galton–Watson processes

We first show that hypersuccinct plane trees achieve an entropy bound in terms of the degree entropy from [60] (see Definition 7.2). In order to prove this entropy bound, we show that hypersuccinct plane trees are universal with respect to the class of Galton–Watson processes (see Definition 4.7).

Recall that for a plane tree $t \in \mathcal{T}$ and a node v of t, we denote with $\deg_t(v)$ the degree of v in t and leave out the subscript t, if the tree t is clear from the context. Moreover, with n_i^t we again denote the number of nodes of degree i of t.

Furthermore, recall the definition of *Galton–Watson processes* from Definition 4.7 in Chapter 4. Let ξ again denote a non-negative integer-valued random variable (the *offspring distribution*) and recall that a Galton–Watson process with offspring distribution ξ assigns a probability $\nu(t)$ to a tree $t \in \mathcal{T}$ by

$$\nu(t) = \prod_{v \in V(t)} \mathbb{P}(\xi = \deg(v)) = \prod_{i \geq 0} \mathbb{P}(\xi = i)^{n_i^t}. \tag{9.1}$$

In order to obtain finite trees with non-zero probability, we should assume that $\mathbb{P}(\xi = 0) > 0$. Moreover, we sometimes use the abbreviation

$$\xi_i = \mathbb{P}(\xi = i).$$

Furthermore, recall the definition of (unnormalized) *degree entropy* from Definition 7.2, which is defined as

$$H^{\deg}(t) = \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right).$$

In the same way as label-shape entropy corresponds to label-shape processes (see Theorem 6.18) and node-type entropy corresponds to node-type processes (see Theorem 8.7), we find that the notion of degree entropy corresponds to Galton–Watson processes. We say that an offspring distribution ξ is the *empirical degree distribution* of a plane tree $t \in \mathcal{T}$, if $\mathbb{P}(\xi = i) = n_i^t/|t|$ for every index $0 \leq i \leq |t|$. In particular, if ξ is the empirical degree distribution of a plane tree $t \in \mathcal{T}$, we have

$$\log\left(\frac{1}{\nu(t)}\right) = \sum_{i=0}^{|t|} n_i^t \log\left(\frac{1}{\mathbb{P}(\xi = i)}\right) = \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right) = H^{\deg}(t).$$

Example 9.4 (Full d-ary trees)**.** Full d-ary trees, that is, trees where each node has either exactly d or 0 children, are obtained from offspring distributions ξ with $\mathbb{P}(\xi = 0) > 0$ and $\mathbb{P}(\xi = d) > 0$ and $\mathbb{P}(\xi = i) = 0$ for $i \neq d, 0$. It is easy to see that a full d-ary tree t with n_d^t many inner nodes (of degree d) always consists of $n_0^t = (d-1)n_d^t + 1$ many leaves, and is thus always of size $d \cdot n_d^t + 1$. The number of full d-ary trees of size $n = dk + 1$, for $k \in \mathbb{N}$, is given by

$$\frac{1}{dk+1}\binom{dk+1}{k}, \tag{9.2}$$

see, e.g., [27]. Let ξ be the offspring distribution with $\mathbb{P}(\xi = 0) = 1/d$ and $\mathbb{P}(\xi = d) = (d-1)/d$. We have

$$\log\left(\frac{1}{\nu(t)}\right) = k \log(d) + \big((d-1)k + 1\big) \log\left(\frac{d}{d-1}\right)$$

for every full d-ary tree t of size $dk + 1$, which is asymptotically, by (9.2), the minimum number of bits needed to represent a full d-ary tree of size $dk + 1$.

Universality with respect to Galton–Watson processes. To prove universality results for hypersuccinct plane trees is conceptually quite similar to the proof techniques used in the previous chapter in order to show universality results for hypersuccinct binary trees.

Given an offspring distribution ξ, Equation (9.1) suggests a way to encode a plane tree $t \in \mathcal{T}$ with $\nu(t) > 0$ in $\log(1/\nu(t))$ (plus lower-order terms) many bits. Such an encoding may spend $\log(1/\mathbb{P}(\xi = i))$ many bits per node v of t of degree $\deg(v) = i$. We use again *arithmetic coding* to encode the degree of node v in that many bits. However, ξ can possibly consist of countably many positive coefficients, thus, we have to adapt the process of arithmetic coding slightly. In order to encode the degree $\deg(v) \in \mathbb{N}_0$ of a node v, we consider $\deg(v)$ as a unary string $s = 0^{\deg(v)}1$, which we encode using arithmetic coding as follows. In order to encode the k^{th} symbol of s, we feed the arithmetic coder with the model that the next symbol is a number $s[k] \in \{0, 1\}$, the probability for $s[k] = 1$ being $\xi_{k-1}/(\xi_{k-1} + \xi_k + \xi_{k+1} + \dots)$ (recall that $\xi_i = \mathbb{P}(\xi = i)$). Thus, arithmetic

coding uses

$$\begin{aligned}
&\sum_{k=0}^{\deg(v)-1} \log\left(\left(1-\frac{\xi_k}{\sum_{i\geq k}\xi_i}\right)^{-1}\right) + \log\left(\frac{\sum_{i\geq \deg(v)}\xi_i}{\xi_{\deg(v)}}\right)\\
&= \sum_{k=0}^{\deg(v)-1} \left(\log\left(\sum_{i\geq k}\xi_i\right) - \log\left(\sum_{i\geq k+1}\xi_i\right)\right) + \log\left(\sum_{i\geq \deg(v)}\xi_i\right) + \log\left(\frac{1}{\xi_{\deg(v)}}\right)\\
&= \log\left(\frac{1}{\xi_{\deg(v)}}\right)
\end{aligned}$$

many bits to encode $s = 0^{\deg(v)}1$. An encoding D_ξ, dependent on a given offspring distribution ξ, stores a tree t as follows. While traversing the tree in depth-first order, we encode the degree $\deg(v)$ of each node v, using arithmetic encoding as described above. We can reconstruct the tree t recursively from its code $D_\xi(t)$, as we always know the degrees of the nodes we have already visited in the depth-first order traversal of the tree. As arithmetic encoding needs $\log(1/\xi_{\deg(v)})$ bits per node v, plus at most 2 bits of overhead, the total number of bits needed in order to store a plane tree $t \in \mathcal{T}$ with $\nu(t) > 0$ is thus

$$|D_\xi(t)| \leq \sum_{v\in V(t)} \log\left(\frac{1}{\xi_{\deg(v)}}\right) + 2.$$

If an offspring distribution ξ is the empirical degree distribution of a plane tree t, that is, $\mathbb{P}(\xi = i) = n_i^t/|t|$ for every $i \in \{0,\ldots,t\}$, we find in particular:

$$|D_\xi(t)| \leq \sum_{i=0}^{|t|} n_i^t \log\left(\frac{|t|}{n_i^t}\right) + 2 = H^{\deg}(t) + 2.$$

The encoding D_ξ yields a prefix-free code for the set of plane trees which satisfy $\nu(t) > 0$ with respect to the degree distribution ξ. In order to show that our hypersuccinct code is universal with respect to Galton–Watson processes, we start with the following lemma:

Lemma 9.5. *Let ξ be a degree distribution and let $t \in \mathcal{T}_n$ be a plane tree of size n with $\nu(t) > 0$. Then*

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \log\left(\frac{1}{\nu(t)}\right) + \mathcal{O}\left(\frac{n\log\log n}{\log n}\right)$$

where Ψ is a Huffman code for the sequence of micro trees $\mathfrak{t}_1,\ldots,\mathfrak{t}_m$ from our tree covering scheme.

Proof. Recall that the micro trees $\mathfrak{t}_1,\ldots,\mathfrak{t}_m$ from our tree partitioning scheme for plane trees are pairwise disjoint except for (potentially) sharing a common subtree root and that apart from edges leaving the subtree root, at most one other edge leads to a node outside of the subtree (Lemma 9.2). Thus, there are at most two nodes in each micro tree $\mathfrak{t}_i$, whose degree in $\mathfrak{t}_i$ might not coincide with

their degree in t: The root of $\mathfrak{t}_i$, which we denote with $\pi_{i,1}$, and at most one node $\pi_{i,2}$ which is not the root node. In particular, for every node $v \neq \pi_{i,1}, \pi_{i,2}$ of $\mathfrak{t}_i$, we have $\deg_{\mathfrak{t}_i}(v) = \deg_t(v)$. Let $\mathrm{pos}(\pi_{i,2})$ denote the depth-first order position of $\pi_{i,2}$ in $\mathfrak{t}_i$. With $D_\xi(\mathfrak{t}_i \setminus \pi_{i,1})$ (respectively, $D_\xi(\mathfrak{t}_i \setminus \pi_{i,1}, \pi_{i,2})$), we denote the following modification of D_ξ. While traversing the tree $\mathfrak{t}_i$ in depth-first order, we encode the degree $\deg_{\mathfrak{t}_i}(v)$ of each node v of $\mathfrak{t}_i$, using arithmetic coding as in the encoding D_ξ, except that we skip the root $\pi_{i,1}$ of $\mathfrak{t}_i$ (respectively, we skip the nodes $\pi_{i,1}$ and $\pi_{i,2}$).

This is well-defined: We have $\xi_{\deg(v)} > 0$ for every node $v \neq \pi_{i,1}, \pi_{i,2}$ of $\mathfrak{t}_i$ whose degree we encode, as $\nu(t) > 0$. If we know $\deg_{\mathfrak{t}_i}(\pi_{i,1})$, respectively, $\deg_{\mathfrak{t}_i}(\pi_{i,1})$, $\deg_{\mathfrak{t}_i}(\pi_{i,2})$ and $\mathrm{pos}(\pi_{i,2})$, we are able to recover $\mathfrak{t}_i$ from $D_\xi(\mathfrak{t}_i \setminus \pi_{i,1})$, respectively, $D_\xi(\mathfrak{t}_i \setminus \pi_{i,1}, \pi_{i,2})$. Let $\mathcal{I}_0$ denote the set of indexes $i \in \{1, \ldots, m\}$ for which $\mathfrak{t}_i$ does not contain a node other than (possibly) the root node from which an edge to a node outside of $\mathfrak{t}_i$ emerges, and let $\mathcal{I}_1 = \{1, \ldots, m\} \setminus \mathcal{I}_0$. We define the following modified encoding:

$$\tilde{D}_\xi(\mathfrak{t}_i) = \begin{cases} 0 \cdot \gamma(\deg_{\mathfrak{t}_i}(\pi_{i,1})) \cdot D_\xi(\mathfrak{t}_i \setminus \pi_{i,1}) & \text{if } i \in \mathcal{I}_0, \\ 1 \cdot \gamma(\deg_{\mathfrak{t}_i}(\pi_{i,1})) \cdot \gamma(\deg_{\mathfrak{t}_i}(\pi_{i,2}) + 1) & \\ \quad \cdot \gamma(\mathrm{pos}(\pi_{i,2})) \cdot D_\xi(\mathfrak{t}_i \setminus \pi_{i,1}, \pi_{i,2}) & \text{otherwise.} \end{cases}$$

Note that formally, $\tilde{D}_\xi$ is again *not* a prefix-free code over $\Sigma_{s_{\max}}$, as there can be micro tree shapes that are assigned *several* codewords by $\tilde{D}_\xi$. But $\tilde{D}_\xi$ can again be seen as a *generalized prefix-free code*, where more than one codeword per symbol is allowed, as $\tilde{D}_\xi$ is uniquely decodable to local shapes of micro trees. Thus, as a Huffman code minimizes the encoding length over the class of *generalized* prefix-free codes, we find:

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \sum_{i=1}^{m} |\tilde{D}_\xi(\mathfrak{t}_i)| = \sum_{i \in \mathcal{I}_0} |\tilde{D}_\xi(\mathfrak{t}_i)| + \sum_{i \in \mathcal{I}_1} |\tilde{D}_\xi(\mathfrak{t}_i)|.$$

We obtain

$$\begin{aligned} \sum_{i \in \mathcal{I}_0} |\tilde{D}_\xi(\mathfrak{t}_i)| &\leq \sum_{i \in \mathcal{I}_0} (|D_\xi(\mathfrak{t}_i \setminus \pi_{i,1})| + 2 \log s_{\max} + 2) \\ &\leq \sum_{i \in \mathcal{I}_0} \Bigg(\sum_{\substack{v \in V(\mathfrak{t}_i) \\ v \neq \pi_{i,1}}} \log\left(\frac{1}{\xi_{\deg_{\mathfrak{t}_i}(v)}}\right) + 2 \log s_{\max} + 4 \Bigg) \end{aligned}$$

as $\deg_{\mathfrak{t}_i}(\pi_{i,1}) \leq s_{\max}$. In the same way, we have

$$\begin{aligned} \sum_{i \in \mathcal{I}_1} |\tilde{D}_\xi(\mathfrak{t}_i)| &\leq \sum_{i \in \mathcal{I}_1} (|D_\xi(\mathfrak{t}_i \setminus \pi_{i,1}, \pi_{i,2})| + 6 \log s_{\max} + 4) \\ &\leq \sum_{i \in \mathcal{I}_1} \Bigg(\sum_{\substack{v \in V(\mathfrak{t}_i) \\ v \neq \pi_{i,1}, \pi_{i,2}}} \log\left(\frac{1}{\xi_{\deg_{\mathfrak{t}_i}(v)}}\right) + 6 \log s_{\max} + 6 \Bigg), \end{aligned}$$

as $\deg_{\mathsf{t}_i}(\pi_{i,2}) + 1, \mathrm{pos}(\pi_{i,2}) \leq s_{\max}$. Since $|\mathcal{I}_0| + |\mathcal{I}_1| = m$, and as every node v of t which is not the root node of a micro tree t_i is contained in at most one subtree t_i and as $\deg_{\mathsf{t}_i}(v) = \deg_t(v)$ for every node $v \neq \pi_{i,1}, \pi_{i,2}$, we have

$$\begin{aligned}\sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| &\leq \sum_{v \in V(t)} \log\left(\frac{1}{\xi_{\deg_t(v)}}\right) + 6m \log s_{\max} + 6m \\ &= \log\left(\frac{1}{\nu(t)}\right) + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right),\end{aligned}$$

as $m = \Theta(n/\log n)$ and $s_{\max} = \Theta(\log n)$. This finishes the proof. □

The following result now follows from Lemma 9.5 and Lemma 9.3:

Theorem 9.6. *Let ξ be an offspring distribution of a Galton–Watson process. The hypersuccinct code $\mathsf{H}\colon \mathcal{T} \to \{0,1\}^*$ satisfies*

$$|\mathsf{H}(t)| \leq \log\left(\frac{1}{\nu(t)}\right) + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right)$$

for every $t \in \mathcal{T}_n$ with $\nu(t) > 0$. In particular, if ξ coincides with the empirical degree distribution of t, we have

$$|\mathsf{H}(t)| \leq H^{\deg}(t) + \mathcal{O}\left(\frac{n \log\log n}{\log n}\right).$$

In particular, we obtain the following corollary from Theorem 9.6:

Corollary 9.7. *The hypersuccinct code $\mathsf{H}\colon \mathcal{T} \to \{0,1\}^*$ optimally encodes full d-ary trees t of size $n = dk + 1$, drawn uniformly at random from the set of all full d-ary trees of size n, using*

$$|\mathsf{H}(t)| \leq k \log(d) + (d-1)k \log(d/(d-1)) + \mathcal{O}(n \log\log n/\log n)$$

many bits.

9.4 Label-shape entropy bound for hypersuccinct trees

Finally, in this section, we show that hypersuccinct plane trees achieve an entropy bound in terms of the label-shape entropy. Recall the definition of the label-shape entropy $H_k^{\ell s}$ from the previous chapter (Definition 6.14 and Definition 6.22) and recall the definition of label-shape histories and label-shape processes, see Section 6.4. Moreover, recall that all these concepts are suitable for *unlabeled* trees as well (that is, to be precise, even though we talk of "label-shape" entropy/histories/processes in the following, no node-labels will be involved). In particular, we thus identify k-label-shape histories with strings $z \in \{0,1\}^k$ and

label-shape processes with tuples $\mathcal{P} = (P_z)_{z\in\{0,1\}^k}$, such that $P_z\colon \{0,2\} \to [0,1]$ is a probability mass function for every $z \in \{0,1\}^k$ in this chapter.

In the following, we show that the length of our hypersuccinct code H for binary trees can be upper-bounded in terms of the k^{th}-order label-shape entropy $H_k^{\ell s}$ of a plane tree (for suitable k), plus lower-order terms. We need some additional notation. We introduce an additional type of tree processes to apply our proof template for universality, which we call *childtype-processes*. The childtype-processes will allow us to write $H_k^{\ell s}(t)$ as $\log\left(1/\mathsf{Prob}_{\mathcal{P}}(t)\right)$, where $\mathsf{Prob}_{\mathcal{P}}(t)$ is the probability that a childtype process $\mathcal{P}$ generates t, which then can be written as a product of contributions of the nodes of the tree.

For an inner node v of a full binary tree $t \in \mathcal{B}$, we define its *childtype* as:

$$\mathsf{childtype}(v) = \begin{cases} 0 & \text{if } v\text{'s children are both leaves,} \\ 1 & \text{if only } v\text{'s left child is a leaf,} \\ 2 & \text{if only } v\text{'s right child is a leaf,} \\ 3 & \text{if } v\text{'s children are both inner nodes.} \end{cases}$$

Moreover, recall the definition of the *first-child next-sibling encoding* from Definition 2.13: the left child (resp. right child) of a node in $\mathrm{fcns}(f)$ is its first child (resp. next sibling) in f or a newly-added leaf, if it does not exist. In particular, we find that $\mathrm{fcns}(f)$ is always a full binary tree, and that $\mathrm{fcns}\colon \mathcal{F} \to \mathcal{B}$ is a bijection. Furthermore, we find that each node v of a forest $f \in \mathcal{F}$ uniquely corresponds to an inner node of $\mathrm{fcns}(f)$, which we denote with $\mathrm{fcns}(v)$. For a node v of a forest $f \in \mathcal{F}$, we set $\mathsf{childtype}(v) = \mathsf{childtype}(\mathrm{fcns}(v))$. In particular, we find:

Lemma 9.8. *Let v be a node of a forest $f \in \mathcal{F}$, then*

$$\mathsf{childtype}(v) = \begin{cases} 0 & \textit{if } v \textit{ is a leaf and does not have a next sibling,} \\ 1 & \textit{if } v \textit{ is a leaf and has a next sibling,} \\ 2 & \textit{if } v \textit{ is not a leaf and does not have a next sibling,} \\ 3 & \textit{if } v \textit{ is not a leaf and has a next sibling.} \end{cases}$$

Here, the next sibling of the root node of a tree of the forest f is the root node of the next tree in the sequence, if it exists. The proof of Lemma 9.8 follows immediately from the definitions of the first-child next-sibling encoding and the $\mathsf{childtype}$-mapping. Furthermore, for a node v of a forest $f \in \mathcal{F}$, we define the label-shape history $h^{\ell s}(v)$ as $h^{\ell s}(\mathrm{fcns}(v))$, i.e., as the label-shape history of its corresponding node in $\mathrm{fcns}(f)$.

We find that if v is the root node of the first tree in (the sequence of trees) f, then $h^{\ell s}(v) = \varepsilon$ (the empty string). Otherwise, if v is the first child of a node w of f, then $h^{\ell s}(v) = h^{\ell s}(w)0$ and if v is the next sibling of a node w of f, then $h^{\ell s}(v) = h^{\ell s}(w)1$. Note that basically, for a node v of a forest f, $h^{\ell s}(v)$ represents the numbers of v's left siblings and of v's ancestors' left siblings in unary. Similarly, we define $h_k^{\ell s}(v)$ as $h_k^{\ell s}(\mathrm{fcns}(v))$.

A k^{th}*-order childtype process* $\mathcal{P} = (n_{\mathcal{P}}, (P_z)_{z\in\{0,1\}^k})$ is a tuple of probability mass functions $P_z\colon \{0,1,2,3\} \to [0,1]$ together with a number $n_{\mathcal{P}} \in [0,1]$. A k^{th}-order childtype process $\mathcal{P}$ assigns a probability $\mathsf{Prob}_{\mathcal{P}}$ to a tree $t \in \mathcal{B}$ by

$$\mathsf{Prob}_{\mathcal{P}}(t) = \begin{cases} 1 - n_{\mathcal{P}} & \text{if } |t| = 1, \\ n_{\mathcal{P}} \cdot \prod\limits_{v\in V_0(t)} P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v)) & \text{otherwise.} \end{cases} \tag{9.3}$$

A k^{th}-order childtype process randomly generates a full binary tree $t \in \mathcal{B}$ as follows. With probability $1 - n_{\mathcal{P}}$, t consists of just one node. Otherwise, in a top-down way, we determine for each node v its $\mathsf{childtype}(v) \in \{0,1,2,3\}$, where this decision depends on the k-label-shape history $h_k^{\ell s}(v)$. The probability that a node v is of childtype i is given by $P_{h_k^{\ell s}(v)}(i)$. We add a left child and a right child to the node and if $i = 0$, we (implicitly) mark both of them as leaves, if $i = 1$, we mark the left child as a leaf, if $i = 2$, we mark the right child as a leaf and if $i = 3$, we do not mark the children as leaves. The process then continues at child nodes which are not marked as leaves. For a forest $f \in \mathcal{F}$, we set

$$\mathsf{Prob}_{\mathcal{P}}(f) = \mathsf{Prob}_{\mathcal{P}}(\mathrm{fcns}(f)). \tag{9.4}$$

Thus, via the fcns-encoding, a k^{th}-order childtype process can be seen as a process randomly generating a forest f as follows: With probability $1 - n_{\mathcal{P}}$, the forest is empty. Otherwise, in a top-down left-to-right way, we determine for each node v its $\mathsf{childtype}(v) \in \{0,1,2,3\}$ (i.e., whether this node has a first child and whether this node has a next sibling), where this decision depends on the k-shape-history $h_k^{\ell s}(v)$. If $\mathsf{childtype}(v) = 0$, the process stops at this node. If $\mathsf{childtype}(v) = 1$, then we add a new child node to v's parent (respectively, if v is a root node itself, we add a new tree of size one to the forest), if $\mathsf{childtype}(v) = 2$, we add a new child to v, and if $\mathsf{childtype}(v) = 3$, we add a new child to v and a new child to v's parent node. The process then continues at newly added nodes.

Lemma 9.9. *Let $t \in \mathcal{T}$ be a non-empty plane tree, then*

$$\mathsf{Prob}_{\mathcal{P}}(t) = n_{\mathcal{P}} \cdot \prod_{v\in V(t)} P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v)).$$

Proof. We have

$$\begin{aligned} \mathsf{Prob}_{\mathcal{P}}(t) = \mathsf{Prob}_{\mathcal{P}}(\mathrm{fcns}(t)) &= n_{\mathcal{P}} \cdot \prod_{v\in V_0(\mathrm{fcns}(t))} P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v)) \\ &= n_{\mathcal{P}} \cdot \prod_{v\in V(t)} P_{h_k^{\ell s}(\mathrm{fcns}(v))}(\mathsf{childtype}(\mathrm{fcns}(v))) \\ &= n_{\mathcal{P}} \cdot \prod_{v\in V(t)} P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v)), \end{aligned}$$

by definition of $\mathsf{Prob}_{\mathcal{P}}$ (see (9.3)), the definition of the k-label-shape history and the definition of the mapping $\mathsf{childtype}$. □

Finally, we make the following definition.

Definition 9.10. Let $\mathcal{P} = (P_z)_{z\in\{0,1\}^k}$ be a k^{th}-order label-shape process. The corresponding $(k-1)^{st}$-order childtype process $\widehat{\mathcal{P}} = (n_{\mathcal{P}}, (\widehat{P}_z)_{z\in\{0,1\}^{k-1}})$ is defined by $n_{\mathcal{P}} = P_{0^k}(2)$ and

$$\widehat{P}_z(0) = P_{z0}(0) \cdot P_{z1}(0), \quad \widehat{P}_z(1) = P_{z0}(0) \cdot P_{z1}(2),$$
$$\widehat{P}_z(2) = P_{z0}(2) \cdot P_{z1}(0), \quad \widehat{P}_z(3) = P_{z0}(2) \cdot P_{z1}(2),$$

for every $z \in \{0,1\}^{k-1}$.

It is easy to see that $\widehat{\mathcal{P}}$ is well-defined. In particular, we find

Lemma 9.11. *Let $t \in \mathcal{B}$ be a full binary tree. Then*

$$\mathsf{Prob}_{\mathcal{P}}(t) = \mathsf{Prob}_{\widehat{\mathcal{P}}}(t).$$

Proof. First, let $|t| = 1$. Then t consists of only one leaf node v of k-label-shape history 0^k, and thus, we have

$$\mathsf{Prob}_{\mathcal{P}}(t) = P_{0^k}(0) = 1 - n_{\mathcal{P}} = \mathsf{Prob}_{\widehat{\mathcal{P}}}(t).$$

In the next part of the proof, assume that $|t| > 1$. Let $\tilde{m}^t_{z,i}$ denote the number of inner nodes of t with k-label-shape history $z \in \{0,1\}^*$ and of childtype $i \in \{0,1,2,3\}$, and recall that $m^t_{z,i}$ denotes the number of nodes of t with k-label-shape-history z and with $\mathsf{type}(v) = i \in \{0,2\}$. Let v be a node of t. First, we assume that $h^{\ell s}_k(v) = z0$ for some $z \in \{0,1\}^{k-1}$ with $z \neq 0^{k-1}$ (thus, v is not the root node of t), and that v is a leaf. Then v's parent w is of $k-1$-label-shape history z, and w's childtype is either 0 or 1. In particular, the correspondence between leaves v of t with k-label-shape history $z0$ and inner nodes $w = \mathrm{parent}(v)$ of t with $k-1$-label-shape-history z and childtype 0 or 1 is bijective, as every node v with k-label-shape history $z0$ is a left child of its parent node. We thus have

$$m^t_{z0,0} = \tilde{m}^t_{z,0} + \tilde{m}^t_{z,1}.$$

In a similar way, we find that inner nodes v of t with k-label-shape history $z0$ for $z \neq 0^{k-1}$ correspond to inner nodes $w = \mathrm{parent}(v)$ of t with $k-1$-label-shape-history z and childtype $i \in \{2,3\}$. We find

$$m^t_{z0,2} = \tilde{m}^t_{z,2} + \tilde{m}^t_{z,3}.$$

Furthermore, we obtain the following relations in the same way:

$$m^t_{z1,0} = \tilde{m}^t_{z,0} + \tilde{m}^t_{z,2},$$
$$m^t_{z1,2} = \tilde{m}^t_{z,1} + \tilde{m}^t_{z,3},$$

for every $z \in \{0,1\}^k$. It remains to deal with nodes of k-label-shape history

$z = 0^k$. We find that every inner node v of t of k-label-shape-history 0^k uniquely corresponds to an inner node $w = \text{parent}(v)$ of t of $(k-1)$-label-shape history 0^{k-1} and childtype $i \in \{2, 3\}$, except for the root node. We thus have

$$m^t_{0^k,2} - 1 = \tilde{m}^t_{0^{k-1},2} + \tilde{m}^t_{0^{k-1},3}.$$

Finally, every leaf v of t of k-label-shape history 0^k uniquely corresponds to an inner node $w = \text{parent}(v)$ of t of $k-1$-label-shape history 0^{k-1} and childtype $i \in \{1, 2\}$, as the root node is an inner node by assumption:

$$m^t_{0^k,0} = \tilde{m}^t_{0^{k-1},0} + \tilde{m}^t_{0^{k-1},1}.$$

Altogether, we thus have for trees t with $|t| > 1$:

$$\begin{aligned}
\mathsf{Prob}_{\mathcal{P}}(t) &= \prod_{z\in\{0,1\}^k} \prod_{i\in\{0,2\}} P_z(i)^{m^t_{z,i}} \\
&= P_{0^k}(0)^{\tilde{m}^t_{0^{k-1},0}+\tilde{m}^t_{0^{k-1},1}} \cdot P_{0^k}(2)^{\tilde{m}^t_{0^{k-1},2}+\tilde{m}^t_{0^{k-1},3}+1} \\
&\quad \cdot \prod_{\substack{z\in\{0,1\}^{k-1} \\ z\neq 0^{k-1}}} P_{z0}(0)^{\tilde{m}^t_{z,0}+\tilde{m}^t_{z,1}} \cdot P_{z0}(2)^{\tilde{m}^t_{z,2}+\tilde{m}^t_{z,3}} \\
&\quad \cdot \prod_{z\in\{0,1\}^{k-1}} P_{z1}(0)^{\tilde{m}^t_{z,0}+\tilde{m}^t_{z,2}} \cdot P_{z1}(2)^{\tilde{m}^t_{z,1}+\tilde{m}^t_{z,3}} \\
&= P_{0^k}(2) \cdot \prod_{z\in\{0,1\}^{k-1}} (P_{z0}(0) \cdot P_{z1}(0))^{\tilde{m}^t_{z,0}} \cdot \prod_{z\in\{0,1\}^{k-1}} (P_{z0}(0) \cdot P_{z1}(2))^{\tilde{m}^t_{z,1}} \\
&\quad \cdot \prod_{z\in\{0,1\}^{k-1}} (P_{z0}(2) \cdot P_{z1}(0))^{\tilde{m}^t_{z,2}} \cdot \prod_{z\in\{0,1\}^{k-1}} (P_{z0}(2) \cdot P_{z1}(2))^{\tilde{m}^t_{z,3}} \\
&= n_{\mathcal{P}} \cdot \prod_{z\in\{0,1\}^{k-1}} \widehat{P}_z(0)^{\tilde{m}^t_{z,0}} \cdot \prod_{z\in\{0,1\}^{k-1}} \widehat{P}_z(1)^{\tilde{m}^t_{z,1}} \\
&\quad \cdot \prod_{z\in\{0,1\}^{k-1}} \widehat{P}_z(2)^{\tilde{m}^t_{z,2}} \cdot \prod_{z\in\{0,1\}^{k-1}} \widehat{P}_z(3)^{\tilde{m}^t_{z,3}} = \mathsf{Prob}_{\widehat{\mathcal{P}}}(t).
\end{aligned}$$

This finishes the proof. □

Corollary 9.12. *Let $t \in \mathcal{T}$ be a plane tree, and let $\mathcal{P}^t := \mathcal{P}^{\text{fcns}(t)}$ denote the empirical label-shape process of its corresponding first-child next-sibling encoding. Then*

$$H^{\ell s}_k(t) = \log\left(\frac{1}{\mathsf{Prob}_{\widehat{\mathcal{P}}^t}(t)}\right).$$

Proof. We have

$$\begin{aligned}
H^{\ell s}_k(t) = H^{\ell s}_k(\text{fcns}(t)) &= \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}^t}(\text{fcns}(t))}\right) \\
&= \log\left(\frac{1}{\mathsf{Prob}_{\widehat{\mathcal{P}}^t}(\text{fcns}(t))}\right) = \log\left(\frac{1}{\mathsf{Prob}_{\widehat{\mathcal{P}}^t}(t)}\right),
\end{aligned}$$

where the first equality follows from Definition 6.22, the second equality follows from the fact that $\mathcal{P}^t$ is the empirical k^{th} order label-shape-process of fcns(t) and Theorem 6.18, the third equality follows from Lemma 9.11 and the last equality follows from (9.4). □

Universality with respect to child-type processes. We apply the same proof template that we have already used for the other universality results of hypersuccinct (plane or binary) trees. In order to show that our hypersuccinct encoding achieves an entropy bound in terms of the label-shape entropy for plane (unlabeled) trees, we start with defining a *source-specific* encoding (again called depth-first order arithmetic code) with respect to a given k^{th}-order childtype process $\mathcal{P}$, against which we will compare the hypersuccinct code.

The formula for $\mathsf{Prob}_{\mathcal{P}}(t)$ from Lemma 9.9 suggests a route for an (essentially) optimal source-specific encoding of any plane tree $t \in \mathcal{T}$, that, given a k^{th} order childtype process $\mathcal{P}$, spends $\log(1/\mathsf{Prob}_{\mathcal{P}}(t))$ (plus lower-order terms) many bits in order to encode a plane tree $t \in \mathcal{T}$ with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. Such an encoding may spend $\log\Big(1/P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v))\Big)$ many bits per node v of t, plus $\log\left(1/n_{\mathcal{P}}\right)$ many bits, if t is non-empty, respectively, $\log\left(1/(1-n_{\mathcal{P}})\right)$ many bits, if t is the empty tree. Assuming that we know the childtype process $\mathcal{P} = (P_z)_{z\in\{0,1\}^k}$, i.e., that we need not store it as part of the encoding, we can make use of *arithmetic coding* again in order to devise a simple (source-dependent) encoding $\mathcal{D}_{\mathcal{P}}$, dependent on $\mathcal{P}$, that stores an plane tree t as follows.

First, we store a number $i \in \{1,2\}$ which tells us whether t is empty ($i = 1$) or non-empty ($i = 2$) using arithmetic encoding, i.e., we feed the arithmetic coder with the model that the first symbol is a number $i \in \{1,2\}$ with probability $1 - n_{\mathcal{P}}$, respectively, $n_{\mathcal{P}}$.

Next, while traversing the tree in depth-first order, we encode for each node v of t its $\mathsf{childtype}(v) \in \{0,1,2,3\}$, using arithmetic coding. To encode $\mathsf{childtype}(v)$ (i.e., whether v is a leaf or not and whether v has a next sibling or not, see Lemma 9.8), we feed the arithmetic coder with the model that the next symbol is a number $i \in \{0,1,2,3\}$ with probability $P_{h_k^{\ell s}(v)}(i)$. Note that we always know $h_k^{\ell s}(v)$ at each node v we traverse. Altogether, this yields a source dependent code $\mathcal{D}_{\mathcal{P}}(t)$, which we refer to as the *depth-first arithmetic code* with respect to the childtype-process $\mathcal{P}$. Note that a plane tree t is always uniquely decodable from $\mathcal{D}_{\mathcal{P}}(t)$. As arithmetic coding uses at most $\log\big(1/P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v))\big)$ many bits per node v, plus $\log\left(1/n_{\mathcal{P}}\right)$ many bits if t is non-empty, plus at most 2 bits of overhead, we find

$$|\mathcal{D}_{\mathcal{P}}(t)| \leq \begin{cases} \displaystyle\sum_{v\in V(t)} \log\left(\frac{1}{P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v))}\right) + \log\left(\frac{1}{n_{\mathcal{P}}}\right) + 2 & \text{if } |t| > 0, \\ \log\left(\dfrac{1}{1-n_{\mathcal{P}}}\right) + 2 & \text{otherwise.} \end{cases}$$

We obtain the following lemma.

Lemma 9.13. *Let $\mathcal{P} = (P_z)_{z\in\{0,1\}^k}$ be a k^{th} order childtype process and let $t \in \mathcal{T}$ be a plane tree of size n with $\mathsf{Prob}_{\mathcal{P}}(t) > 0$. Then*

$$\sum_{i=1}^{m} |\Psi(\mathfrak{t}_i)| \leq \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}\left(\frac{n\log\log n + kn}{\log n}\right),$$

where Ψ is a Huffman code for the sequence of micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ obtained from the tree-covering scheme.

Proof. Recall that the micro trees $\mathfrak{t}_1, \ldots, \mathfrak{t}_m$ from our tree partitioning scheme for plane trees are pairwise disjoint except for (potentially) sharing a common subtree root and that apart from edges leaving the subtree root, at most one other edge leads to a node outside of the subtree (see Lemma 9.2). The probability $\mathsf{Prob}_{\mathcal{P}}(t)$ consists of the contributions $P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v))$ for every node v of t. However, $P_{h_k^{\ell s}(v)}(\mathsf{childtype}(v))$ depends on the childtype and k-label-shape history of each node v, and there might be nodes, for which childtype and k-label-shape history differ in t and $\mathfrak{t}_i$.

For the sake of clarity, let $h_k^{\ell s}(v,t)$ denote the k-label-shape history of a node v in t (and likewise $h_k^{\ell s}(v,\mathfrak{t}_i)$ the k-label-shape history of a node v in a micro tree $\mathfrak{t}_i$), and let $\mathsf{childtype}(v,t)$ (resp. $\mathsf{childtype}(v,\mathfrak{t}_i)$) denote the childtype of a node v in t (resp. $\mathfrak{t}_i$). First, we investigate under which conditions it might occur that a node v of micro tree $\mathfrak{t}_i$ satisfies $h_k^{\ell s}(v,t) \neq h_k^{\ell s}(v,\mathfrak{t}_i)$ or $\mathsf{childtype}(v,t) \neq \mathsf{childtype}(v,\mathfrak{t}_i)$. We find:

(i) If v is the root node of a micro tree $\mathfrak{t}_i$, then it might have left, respectively, right siblings in t, which it does not have in $\mathfrak{t}_i$. Thus, its childtype and its k-label-shape history might change.

(ii) If v is the first child of the root of $\mathfrak{t}_i$, then it might have left siblings in t, which it does not have in $\mathfrak{t}_i$. Thus, its k-label-shape history changes. Furthermore, the k-label-shape history of its close descendants and right siblings thus changes as well, i.e., the k-label-shape history of the descendants of order less than k of $\mathrm{fcns}(v)$. However, if we know $h_k^{\ell s}(v,t)$, we are able to recover $h_k^{\ell s}(w,t)$ for all nodes w which are descendants, right siblings, or right siblings of descendants of v.

(iii) If v is the last child of the root of $\mathfrak{t}_i$, then it might have right siblings in t, which it does not have in $\mathfrak{t}_i$. Thus, its childtype might change.

(iv) The root node's children in $\mathfrak{t}_i$ are consecutive children of this node in t, except for possibly one child node x, which might be missing in $\mathfrak{t}_i$ (see Lemma 9.2). Thus, if v is the right sibling of x in t, its k-label-shape history in $\mathfrak{t}_i$ might differ from its k-label-shape history in t. Furthermore, the k-label-shape histories of nodes corresponding to the descendants of order at most k of $\mathrm{fcns}(v)$ in $\mathrm{fcns}(t)$ might change as well. Again, if we know $h_k^{\ell s}(v,t)$, we are able to recover $h_k^{\ell s}(w,t)$ of nodes w which correspond to descendants of $\mathrm{fcns}(v)$ in $\mathrm{fcns}(t)$.

(v) There is at most one other edge which leads to a node outside of the micro tree $\mathfrak{t}_i$, besides edges emanating from the root of $\mathfrak{t}_i$ (by Lemma 9.2). Let v be the node in $\mathfrak{t}_i$, from which this other edge emanates. If v has only one child in t, then it does not have a child node in $\mathfrak{t}_i$, and thus, its childtypes in t and $\mathfrak{t}_i$ do not coincide. Otherwise, the degree of v in t is greater than one and in particular, there might be a child node w of v, whose left sibling in t does not belong to $\mathfrak{t}_i$. Thus, w's k-label-shape history might change, as well as the k-label-shape history of the nodes corresponding to the descendants of order less than k of $\mathrm{fcns}(w)$. Again, if we know $h_k^{\ell s}(v,t)$, we are able to recover $h_k^{\ell s}(w,t)$ of nodes w which correspond to descendants of $\mathrm{fcns}(v)$ in $\mathrm{fcns}(t)$. Finally, there might be a child node u of v, which has a right sibling in t and which does not have a right sibling in $\mathfrak{t}_i$; thus, its childtype changes.

By the above considerations, there can be several nodes v in $\mathfrak{t}_i$ for which $h_k^{\ell s}(v,t) \neq h_k^{\ell s}(v,\mathfrak{t}_i)$, however, we only need to know $h_k^{\ell s}(v,t)$ for at most four of these nodes (see items (i), (ii), (iv) and (v)) in order to be able to determine the k-label-shape history in t of all nodes of $\mathfrak{t}_i$. Let n_i denote the number of k-label-shape histories we need to know in order to be able to determine $h_k^{\ell s}(v,t)$ for all nodes v of $\mathfrak{t}_i$. Furthermore, let j_i denote the number of nodes v of $\mathfrak{t}_i$, for which $\mathsf{childtype}(v,t) \neq \mathsf{childtype}(v,\mathfrak{t})$, where we always include the root node π_i of $\mathfrak{t}_i$ in this j_i many nodes (even if its childtypes in t and $\mathfrak{t}_i$ are identical). By the above considerations, we find that j_i is upper-bounded by four (see items (i), (iii) and (v)). Let $S_i \in \{0,1\}^*$ denote the following binary string, obtained as the concatenation of

- an encoding of the number j_i using two bits,
- the preorder positions in $\mathfrak{t}_i$ of the j_i many nodes for which it holds that $\mathsf{childtype}(v,t) \neq \mathsf{childtype}(v,\mathfrak{t}_i)$ (plus the root node π_i of $\mathfrak{t}_i$), encoded in Elias gamma code and listed in preorder,
- the encodings of the childtypes in t of these j_i nodes using two bits each, listed in preorder,
- the encodings of the childtypes in $\mathfrak{t}_i$ of these j_i nodes using two bits each, listed in preorder,
- an encoding of the number n_i using two bits,
- the Elias gamma encodings of the preorder positions in $\mathfrak{t}_i$ of the n_i many nodes from whose k-label-shape histories in t we are able to determine the k-label-shape history in t of all nodes of $\mathfrak{t}_i$, listed in preorder,
- the k-label-shape histories of these n_i nodes, listed in preorder, using k bits each.

We find that $|S_i| \leq \mathcal{O}(\log(s_{\max}) + k)$. We define the following modification of the depth-first order arithmetic code $\mathcal{D}_{\mathcal{P}}$, which we denote with $\bar{\mathcal{D}}_{\mathcal{P}}$. The encoding

$\bar{\mathcal{D}}_{\mathcal{P}}(\mathsf{t}_i)$ consists of the string S_i followed by an encoding of $\mathsf{childtype}(v,t)$ for every node v of t_i in depth-first order (preorder) of t_i except for the root node π_i of t_i, using arithmetic coding. The childtype of the root node π_i is already stored in S_i. We traverse the tree t_i in depth-first order; to encode $\mathsf{childtype}(v,t)$, we feed the arithmetic coder with the model that the next symbol is a number $i \in \{0,1,2,3\}$ with probability $P_{h_k^{\ell s}(v,t)}(i)$. Note that at each node v that we pass, we know $h_k^{\ell s}(v,t)$ (either from S_i or as we are able to determine $h_k^{\ell s}(v,t)$ from the k-label-shape history of the node v's left sibling or parent) and we know both $\mathsf{childtype}(v,t)$ and $\mathsf{childtype}(v,\mathsf{t}_i)$ (either because $\mathsf{childtype}(v,t) = \mathsf{childtype}(v,\mathsf{t}_i)$ or because we have stored both $\mathsf{childtype}(v,t)$ and $\mathsf{childtype}(v,\mathsf{t}_i)$ explicitly in S_i). Altogether, this yields the encoding $\bar{\mathcal{D}}(\mathsf{t}_i)$. Note that we leave out the $\log(1/n_{\mathcal{P}})$ many bits (used in the encoding $\mathcal{D}(\mathsf{t}_i)$) which encode the number $i \in \{1,2\}$ which tells us whether t_i is empty or not (by definition, every micro tree t_i of a non-empty tree t is non-empty). As we have $P_{h_k^{\ell s}(v,t)}(\mathsf{childtype}(v,t)) > 0$ for every node v the encoding $\bar{\mathcal{D}}_{\mathcal{P}}(\mathsf{t}_i)$ is well-defined. We find that

$$|\bar{\mathcal{D}}_{\mathcal{P}}(\mathsf{t}_i)| \leq |S_i| + \sum_{\substack{v \in \mathsf{t}_i \\ v \neq \pi_i}} \log(1/P_{h_k^{\ell s}(v,t)}(\mathsf{childtype}(v,t))) + 2.$$

Furthermore, note that we can uniquely recover a micro tree shape t_i from the encoding $\bar{\mathcal{D}}_{\mathcal{P}}(\mathsf{t}_i)$ and that formally, $\bar{\mathcal{D}}_{\mathcal{P}}$ is not a *prefix-free* code over $\Sigma_{s_{\max}}$, as there can be micro tree shapes that are assigned several codewords by $\bar{\mathcal{D}}_{\mathcal{P}}$. But $\bar{\mathcal{D}}_{\mathcal{P}}$ can again be seen as a generalized prefix-free code, where more than one codeword per symbol is allowed, as $\bar{\mathcal{D}}_{\mathcal{P}}$ is uniquely decodable to local shapes of micro trees. Thus, as a Huffman code minimizes the encoding length over the class of generalized prefix-free codes, we find:

$$\begin{aligned}\sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| &\leq \sum_{i=1}^{m} |\bar{\mathcal{D}}_{\mathcal{P}}(\mathsf{t}_i)| \\ &\leq \sum_{i=1}^{m} \Bigg(|\mathcal{S}_i| + \sum_{\substack{v \in V(\mathsf{t}_i) \\ v \neq \pi_i}} \log(1/P_{h_k^{\ell s}(v,t)}(\mathsf{childtype}(v,t))) + 2 \Bigg).\end{aligned}$$

Recall that the micro trees t_i are disjoint except for possibly sharing a common root node and that $|S_i| \leq \mathcal{O}(\log s_{\max} + k)$. Thus, we have

$$\sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| \leq \sum_{v \in V(t)} \log(1/P_{h_k^{\ell s}(v,t)}(\mathsf{childtype}(v,t))) + \mathcal{O}(m \log s_{\max} + mk).$$

With $m = \Theta(n/\log n)$ and $s_{\max} = \Theta(\log n)$ (see Section 9.2), we have

$$\sum_{i=1}^{m} |\Psi(\mathsf{t}_i)| \leq \log\left(\frac{1}{\mathsf{Prob}_{\mathcal{P}}(t)}\right) + \mathcal{O}\left(\frac{n \log\log n + kn}{\log n}\right).$$

This finishes the proof. □

From Lemma 9.3, Lemma 9.13 and Corollary 9.12, we obtain the following theorem:

Theorem 9.14. *The hypersuccinct encoding* $\mathsf{H}\colon \mathcal{T} \to \{0,1\}^*$ *satisfies*

$$|\mathsf{H}(t)| \leq H_k^{\ell s}(t) + \mathcal{O}\left(\frac{n \log\log n + kn}{\log n}\right)$$

for every plane tree $t \in \mathcal{T}$ *of size* n.

It remains to remark that the above result requires $k \leq o(\log n)$ in order to be non-trivial. This bound on k also occurs in Theorem 8.14 for the node-type entropy, Theorem 6.21 for the label-shape entropy, respectively, the entropy bounds from [89, 46]. Moreover, note that the redundancy term in Theorem 9.14 is identical to the redundancy terms in Theorem 8.14 from Chapter 8 and Theorem 6.21 and Theorem 6.23 from Chapter 6.

9.5 Conclusion and open problems

Hypersuccinct plane trees achieve an entropy bound both in terms of the label-shape entropy, as well as in terms of the degree entropy. A comparison of label-shape entropy and degree entropy as empirical entropy measures for unlabeled plane trees was shown in Section 7.3, see in particular Theorem 7.12.

Further universality results with respect to hypersuccinct plane trees are presented in [S7]. Specifically, it is shown that the hypersuccinct tree encoding is (worst-case) universal with respect to a subclasses of *fixed-size ordinal tree sources*, which represent a plane-tree-analogue of fixed-size binary tree sources.

A natural open problem is to generalize the hypersuccinct tree encoding from unlabeled plane trees to labeled plane trees. In particular, there is no compressed data structure for labeled plane trees (over a non-unary alphabet) known to achieve an entropy bound in terms of the label-shape entropy and at the same time, to achieve constant query times in the word-RAM model. Moreover, for labeled plane trees the entropy notions from Ganczorz [46] become suitable, such that an interesting task would be to show that hypersuccinct labeled plane trees achieve entropy bounds both in terms of label-shape entropy and in terms of Ganczorz's entropy notions.

Over the last few years, we have seen increasing efforts aiming to generalize aspects of information theory as entropy notions and universal source coding from strings to structured data as trees and graphs; further results include e.g. [79, 49, 18, 77]. However, compared with the situation for strings (see, e.g., [20]), the information theory of structured data is much less developed, leaving many open problems for future research.

Resulting publications

[S1] Jarno N. Alanko, Travis Gagie, Gonzalo Navarro, and Louisa Seelbach Benkner. Tunneling on Wheeler graphs. In *Proceedings of the Data Compression Conference, DCC 2019*, pages 122–131. IEEE, 2019. doi:10.1109/DCC.2019.00020.

[S2] Travis Gagie, Tomohiro I, Giovanni Manzini, Gonzalo Navarro, Hiroshi Sakamoto, Louisa Seelbach Benkner, and Yoshimasa Takabatake. Practical random access to SLP-compressed texts. In *Proceedings of the 27th International String Processing and Information Retrieval Symposium, SPIRE 2020*, volume 12303 of *Lecture Notes in Computer Science*, pages 221–231. Springer, 2020. doi:10.1007/978-3-030-59212-7_16.

[S3] Moses Ganardi, Danny Hucke, Markus Lohrey, and Louisa Seelbach Benkner. Universal tree source coding using grammar-based compression. *IEEE Transactions on Information Theory*, 65(10):6399–6413, 2019. doi:10.1109/TIT.2019.2919829.

[S4] Danny Hucke, Markus Lohrey, and Louisa Seelbach Benkner. Entropy bounds for grammar-based tree compressors. In *Proceedings of the IEEE International Symposium on Information Theory, ISIT 2019*, pages 1687–1691. IEEE, 2019. doi:10.1109/ISIT.2019.8849372.

[S5] Danny Hucke, Markus Lohrey, and Louisa Seelbach Benkner. A comparison of empirical tree entropies. In *Proceedings of the 27th International String Processing and Information Retrieval Symposium, SPIRE 2020*, volume 12303 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 2020. doi:10.1007/978-3-030-59212-7_17.

[S6] Danny Hucke, Markus Lohrey, and Louisa Seelbach Benkner. Entropy bounds for grammar-based tree compressors. *IEEE Transactions on Information Theory*, 67(11):7596–7615, 2021. doi:10.1109/TIT.2021.3112676.

[S7] J. Ian Munro, Patrick K. Nicholson, Louisa Seelbach Benkner, and Sebastian Wild. Hypersuccinct trees - new universal tree source codes for optimal compressed tree data structures and range minima. In *Proceedings of the 29th Annual European Symposium on Algorithms, ESA 2021*, volume 204 of *LIPIcs*, pages 70:1–70:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ESA.2021.70.

[S8] Louisa Seelbach Benkner and Markus Lohrey. Average case analysis of leaf-centric binary tree sources. In *Proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018*, volume 117 of *LIPIcs*, pages 16:1–16:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.16.

[S9] Louisa Seelbach Benkner, Markus Lohrey, and Stephan Wagner. Average case analysis of leaf-centric binary tree sources. *CoRR*, abs/1804.10396v2, 2018. URL: http://arxiv.org/abs/1804.10396v2, arXiv:1804.10396.

[S10] Louisa Seelbach Benkner and Stephan Wagner. On the collection of fringe subtrees in random binary trees. In *Proceedings of the 14th Latin American Symposium on Theoretical Informatics, LATIN 2020*, volume 12118 of *Lecture Notes in Computer Science*, pages 546–558. Springer, 2020. doi:10.1007/978-3-030-61792-9_43.

[S11] Louisa Seelbach Benkner and Stephan Wagner. Distinct fringe subtrees in random trees. *Algorithmica*, 84(12):3686–3728, 2022. doi:10.1007/s00453-022-01013-y.

Bibliography

[1] Serge Abiteboul, Pierre Bourhis, and Victor Vianu. Highly expressive query languages for unordered data trees. *Theory of Computing Systems*, 57(4):927–966, 2015. doi:10.1007/s00224-015-9617-5.

[2] Janos Aczél. On Shannon's inequality, optimal coding, and characterizations of Shannon's and Renyi's entropies. Technical Report AA-73-05, University of Waterloo, 1973. https://cs.uwaterloo.ca/research/tr/1973/CS-73-05.pdf.

[3] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley series in computer science / World student series edition. Addison-Wesley, 1986.

[4] David Aldous. Asymptotic fringe distributions for general families of random trees. *The Annals of Applied Probability*, 1(2):228–266, 1991. doi:10.1214/aoap/1177005936.

[5] Michael Artin. *Algebra*. Pearson Prentice Hall, 2011.

[6] David Benoit, Erik D. Demaine, J. Ian Munro, Rajeev Raman, Venkatesh Raman, and Srinivasa Rao Satti. Representing trees of higher degree. *Algorithmica*, 43(4):275–292, 2005. doi:10.1007/s00453-004-1146-6.

[7] François Bergeron, Philippe Flajolet, and Bruno Salvy. Varieties of increasing trees. In *Proceedings of the 17th Colloquium on Trees in Algebra and Programming 1992, CAAP '92*, volume 581 of *Lecture Notes in Computer Science*, pages 24–48. Springer, 1992. doi:10.1007/3-540-55251-0_2.

[8] Philip Bille, Inge Li Gørtz, Gad M. Landau, and Oren Weimann. Tree compression with top trees. *Information and Computation*, 243:166–177, 2015. doi:10.1016/j.ic.2014.12.012.

[9] Olivier Bodini, Antoine Genitrini, Bernhard Gittenberger, Isabella Larcher, and Mehdi Naima. Compaction for two models of logarithmic-depth trees: Analysis and experiments. *Random Structures and Algorithms*, 61(1):31–61, 2022. doi:10.1002/rsa.21056.

[10] Miklós Bóna and Philippe Flajolet. Isomorphism and symmetries in random phylogenetic trees. *Journal of Applied Probability*, 46(4):1005–1019, 2009. doi:10.1017/s0021900200006100.

[11] Iovka Boneva, Radu Ciucanu, and Slawek Staworko. Schemas for unordered XML on a DIME. *Theory of Compuing Systems*, 57(2):337–376, 2015. doi:10.1007/s00224-014-9593-1.

[12] Mireille Bousquet-Mélou, Markus Lohrey, Sebastian Maneth, and Eric Nöth. XML compression via directed acyclic graphs. *Theory of Computing Systems*, 57(4):1322–1371, 2015. doi:10.1007/s00224-014-9544-x.

[13] Randal E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, 1992. doi:10.1145/136035.136043.

[14] Peter Buneman, Martin Grohe, and Christoph Koch. Path queries on compressed XML. In *Proceedings of the 29th Conference on Very Large Data Bases, VLDB 2003*, pages 141–152. Morgan Kaufmann, 2003. doi:10.1016/B978-012722442-8/50021-5.

[15] Giorgio Busatto, Markus Lohrey, and Sebastian Maneth. Efficient memory representation of XML document trees. *Information Systems*, 33(4–5):456–474, 2008. doi:10.1016/j.is.2008.01.004.

[16] Xing Shi Cai. *A study of large fringe and non-fringe subtrees in conditional Galton-Watson trees*. PhD thesis, McGill University, 2016.

[17] Moses Charikar, Eric Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, Amit Sahai, and Abhi Shelat. The smallest grammar problem. *IEEE Transactions on Information Theory*, 51(7):2554–2576, 2005. doi:10.1109/TIT.2005.850116.

[18] Yongwook Choi and Wojciech Szpankowski. Compression of graphical structures: Fundamental limits, algorithms, and experiments. *IEEE Transactions on Information Theory*, 58(2):620–638, 2012. doi:10.1109/TIT.2011.2173710.

[19] Thomas M. Cover. Enumerative source encoding. *IEEE Transactions on Information Theory*, 19(1):73–77, 1973. doi:10.1109/TIT.1973.1054929.

[20] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley, 2nd edition, 2006. doi:10.1002/0471200611.

[21] Pooya Davoodi, Gonzalo Navarro, Rajeev Raman, and Srinivasa Rao Satti. Encoding range minima and range top-2 queries. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 372(2016):20130131, 2014. doi:10.1098/rsta.2013.0131.

[22] Pooya Davoodi, Rajeev Raman, and Srinivasa Rao Satti. On succinct representations of binary trees. *Mathematics in Computer Science*, 11(2):177–189, 2017. doi:10.1007/s11786-017-0294-4.

[23] Florian Dennert and Rudolf Grübel. On the subtree size profile of binary search trees. *Combinatorics, Probability and Computing*, 19(4):561–578, 2010. doi:10.1017/S0963548309990630.

[24] Luc Devroye. On the richness of the collection of subtrees in random binary search trees. *Information Processing Letters*, 65(4):195–199, 1998. doi:10.1016/S0020-0190(97)00206-8.

[25] Luc Devroye and Svante Janson. Protected nodes and fringe subtrees in some random trees. *Electronic Communications in Probability*, 19:1–10, 2014. doi:10.1214/ECP.v19-3048.

[26] Peter J. Downey, Ravi Sethi, and Robert Endre Tarjan. Variations on the common subexpression problem. *Journal of the ACM*, 27(4):758–771, 1980. doi:10.1145/322217.322228.

[27] Michael Drmota. *Random Trees: An Interplay Between Combinatorics and Probability*. Springer Publishing Company, Incorporated, 1st edition, 2009. doi:10.1007/978-3-211-75357-6.

[28] Michelle Effros, Karthik Visweswariah, Sanjeev R. Kulkarni, and Sergio Verdú. Universal lossless source coding with the Burrows Wheeler transform. *IEEE Transactions on Information Theory*, 48(5):1061–1081, 2002. doi:10.1109/18.995542.

[29] Peter Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21(2):194–203, 1975. doi:10.1109/TIT.1975.1055349.

[30] Arash Farzan and J. Ian Munro. A uniform paradigm to succinctly encode various families of trees. *Algorithmica*, 68(1):16–40, 2014. doi:10.1007/s00453-012-9664-0.

[31] Qunqiang Feng and Hosam M. Mahmoud. On the variety of shapes on the fringe of a random recursive tree. *Journal of Applied Probability*, 47(1):191–200, 2010. doi:10.1239/jap/1269610825.

[32] Paolo Ferragina, Fabrizio Luccio, Giovanni Manzini, and S. Muthukrishnan. Structuring labeled trees for optimal succinctness, and beyond. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2005*, pages 184–196. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.69.

[33] Paolo Ferragina, Fabrizio Luccio, Giovanni Manzini, and S. Muthukrishnan. Compressing and indexing labeled trees, with applications. *Journal of the ACM*, 57(1):4:1–4:33, 2009. doi:10.1145/1613676.1613680.

[34] James Allen Fill and Nevin Kapur. Limiting distributions for additive functionals on Catalan trees. *Theoretical Computer Science*, 326(1-3):69–102, 2004. doi:10.1016/j.tcs.2004.05.010.

[35] Steven R. Finch and Gian-Carlo Rota. *Mathematical Constants*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2003.

[36] Philippe Flajolet, Xavier Gourdon, and Conrado Martínez. Patterns in random binary search trees. *Random Structures & Algorithms*, 11(3):223–244, 1997.

[37] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009. doi:10.1017/cbo9780511801655.

[38] Philippe Flajolet, Paolo Sipala, and Jean-Marc Steyaert. Analytic variations on the common subexpression problem. In *Proceedings of the 17th International Colloquium on Automata, Languages and Programming, ICALP 1990*, volume 443 of *Lecture Notes in Computer Science*, pages 220–234. Springer, 1990. doi:10.1007/BFb0032034.

[39] Markus Frick, Martin Grohe, and Christoph Koch. Query evaluation on compressed trees (extended abstract). In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science, LICS 2003*, pages 188–197. IEEE Computer Society Press, 2003. doi:10.1109/LICS.2003.1210058.

[40] Michael Fuchs. Limit theorems for subtree size profiles of increasing trees. *Combinatorics, Probability and Computing*, 21(3):412–441, 2012. doi:10.1017/S096354831100071X.

[41] Travis Gagie. Large alphabets and incompressibility. *Information Processing Letters*, 99(6):246–251, 2006. doi:10.1016/j.ipl.2006.04.008.

[42] Moses Ganardi, Danny Hucke, Artur Jez, Markus Lohrey, and Eric Noeth. Constructing small tree grammars and small circuits for formulas. *Journal of Computer and System Sciences*, 86:136–158, 2017. doi:10.1016/j.jcss.2016.12.007.

[43] Moses Ganardi, Artur Jez, and Markus Lohrey. Balancing straight-line programs. *Journal of the ACM*, 68(4):27:1–27:40, 2021. doi:10.1145/3457389.

[44] Moses Ganardi and Markus Lohrey. A universal tree balancing theorem. *ACM Transactions on Compututation Theory*, 11(1):1:1–1:25, 2019. doi:10.1145/3278158.

[45] Michal Ganczorz. Entropy bounds for grammar compression. *CoRR*, abs/1804.08547, 2018. URL: http://arxiv.org/abs/1804.08547.

[46] Michal Ganczorz. Using statistical encoding to achieve tree succinctness never seen before. In *Proceedings of the 37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020*, volume 154 of *LIPIcs*, pages 22:1–22:29. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.STACS.2020.22.

[47] Adrià Gascón, Markus Lohrey, Sebastian Maneth, Carl Philipp Reh, and Kurt Sieber. Grammar-based compression of unranked trees. *Theory of Computing Systems*, 64(1):141–176, 2020. doi:10.1007/s00224-019-09942-y.

[48] Richard F. Geary, Rajeev Raman, and Venkatesh Raman. Succinct ordinal trees with level-ancestor queries. *ACM Transactions on Algorithms*, 2(4):510–534, 2006. doi:10.1145/1198513.1198516.

[49] Zbigniew Golebiewski, Abram Magner, and Wojciech Szpankowski. Entropy and optimal compression of some general plane trees. *ACM Transactions on Algorithms*, 15(1):3:1–3:23, 2019. doi:10.1145/3275444.

[50] Mordecai J. Golin, John Iacono, Danny Krizanc, Rajeev Raman, Srinivasa Rao Satti, and Sunil M. Shende. Encoding 2d range maximum queries. *Theoretical Computer Science*, 609:316–327, 2016. doi:10.1016/j.tcs.2015.10.012.

[51] Allan Gut. *Probability: A Graduate Course*. Springer, 2005.

[52] Meng He, J. Ian Munro, and Srinivasa Rao Satti. Succinct ordinal trees based on tree covering. *ACM Transactions on Algorithms*, 8(4):42:1–42:32, 2012. doi:10.1145/2344422.2344432.

[53] Cecilia Holmgren, Svante Janson, and Matas Šileikis. Multivariate normal limit laws for the numbers of fringe subtrees in m-ary search trees and preferential attachment trees. *Electronic Journal of Combinatorics*, 24(2):2, 2017. doi:10.37236/6374.

[54] Lorenz Hübschle-Schneider and Rajeev Raman. Tree compression with top trees revisited. In *Proceedings of the 14th International Symposium on Experimental Algorithms, SEA 15*, volume 9125 of *Lecture Notes in Computer Science*, pages 15–27. Springer, 2015. doi:10.1007/978-3-319-20086-6_2.

[55] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[56] Guy Jacobson. Space-efficient static trees and graphs. In *Proceedings of the 30th Symposium on Foundations of Computer Science, FOCS 1989*, pages 549–554. IEEE Computer Society, 1989. doi:10.1109/SFCS.1989.63533.

[57] Svante Janson. Random cutting and records in deterministic and random trees. *Random Structures & Algorithms*, 29(2):139–179, 2006. doi:10.1002/rsa.20086.

[58] Svante Janson. Simply generated trees, conditioned Galton–Watson trees, random allocations and condensation. *Probability Surveys*, 9:103–252, 2012. doi:10.1214/11-ps188.

[59] Svante Janson. Asymptotic normality of fringe subtrees and additive functionals in conditioned Galton-Watson trees. *Random Structures & Algorithms*, 48(1):57–101, 2016. doi:10.1002/rsa.20568.

[60] Jesper Jansson, Kunihiko Sadakane, and Wing-Kin Sung. Ultra-succinct representation of ordered trees with applications. *Journal of Computer and System Sciences*, 78(2):619–631, 2012. doi:10.1016/j.jcss.2011.09.002.

[61] Camille Jordan. Sur les assemblages de lignes. *Journal für die reine und angewandte Mathematik*, 70:185–190, 1869. doi:10.1515/crll.1869.70.185.

[62] Dominik Kempa and Nicola Prezza. At the roots of dictionary compression: string attractors. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 827–840. ACM, 2018. doi:10.1145/3188745.3188814.

[63] John C. Kieffer, En hui Yang, Gregory J. Nelson, and Pamela C. Cosman. Universal lossless compression via multilevel pattern matching. *IEEE Transactions on Information Theory*, 46(4):1227–1245, 2000. doi:10.1109/18.850665.

[64] John C. Kieffer and En-Hui Yang. Grammar-based codes: A new class of universal lossless source codes. *IEEE Transactions on Information Theory*, 46(3):737–754, 2000. doi:10.1109/18.841160.

[65] John C. Kieffer and En-Hui Yang. Structured grammar-based codes for universal lossless data compression. *Communications in Information and Systems*, 2(1):29–52, 2002. doi:10.4310/cis.2002.v2.n1.a2.

[66] John C. Kieffer, En-Hui Yang, and Wojciech Szpankowski. Structural complexity of random binary trees. In *Proceedings of the IEEE International Symposium on Information Theory, ISIT 2009*, pages 635–639. IEEE, 2009. doi:10.1109/ISIT.2009.5205704.

[67] Donald E. Knuth. *The art of computer programming. Vol. 3.* Addison-Wesley, Reading, MA, 1998.

[68] Tomasz Kociumaka, Gonzalo Navarro, and Nicola Prezza. Toward a definitive compressibility measure for repetitive sequences. *IEEE Transactions on Information Theory*, 69(4):2074–2092, 2023. doi:10.1109/TIT.2022.3224382.

[69] Valentin F. Kolchin. *Random mappings.* Translations series in mathematics and engineering. Optimization Software, New York, 1986.

[70] S. Rao Kosaraju and Giovanni Manzini. Compression of low entropy strings with Lempel-Ziv algorithms. *SIAM Journal on Computing*, 29(3):893–911, 1999. doi:10.1137/S0097539797331105.

[71] Sebastian Kreft and Gonzalo Navarro. On compressing and indexing repetitive sequences. *Theoretical Computer Science*, 483:115–133, 2013. doi:10.1016/j.tcs.2012.02.006.

[72] N. Jesper Larsson and Alistair Moffat. Offline dictionary-based compression. In *Proceedings of the 1999 Data Compression Conference, DCC 1999*, pages 296–305. IEEE Computer Society Press, 1999. doi:10.1109/DCC.1999.755679.

[73] Markus Lohrey. Grammar-based tree compression. In *Proceedings of the 19th International Conference on Developments in Language Theory, DLT 2015*, volume 9168 of *Lecture Notes in Computer Science*, pages 46–57. Springer, 2015. doi:10.1007/978-3-319-21500-6_3.

[74] Markus Lohrey, Sebastian Maneth, and Roy Mennicke. XML tree structure compression using RePair. *Information Systems*, 38(8):1150–1167, 2013. doi:10.1016/j.is.2013.06.006.

[75] Markus Lohrey, Sebastian Maneth, and Carl Philipp Reh. Compression of unordered XML trees. In *Proceedings of the 20th International Conference on Database Theory, ICDT 2017*, volume 68 of *LIPIcs*, pages 18:1–18:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICDT.2017.18.

[76] Markus Lohrey, Carl Philipp Reh, and Kurt Sieber. Size-optimal top dag compression. *Information Processing Letters*, 147:27–31, 2019. doi:10.1016/j.ipl.2019.03.001.

[77] Tomasz Luczak, Abram Magner, and Wojciech Szpankowski. Compression of preferential attachment graphs. In *Proceedings of the IEEE International Symposium on Information Theory, ISIT 2019*, pages 1697–1701. IEEE, 2019. doi:10.1109/ISIT.2019.8849739.

[78] Mokshay Madiman. On the entropy of sums. In *Proceedings of the 2008 IEEE Information Theory Workshop, ITW 2008*, pages 303–307. IEEE, 2008. doi:10.1109/ITW.2008.4578674.

[79] Abram Magner, Krzysztof Turowski, and Wojciech Szpankowski. Lossless compression of binary trees with correlated vertex names. *IEEE Transactions on Information Theory*, 64(9):6070–6080, 2018. doi:10.1109/TIT.2018.2851224.

[80] Giovanni Manzini. An analysis of the Burrows-Wheeler transform. *Journal of the ACM*, 48(3):407–430, 2001. doi:10.1145/382780.382782.

[81] Conrado Martínez. *Statistics under the BST model.* PhD thesis, University Barcelona, 1992.

[82] Amram Meir and John W. Moon. On the altitude of nodes in random trees. *Canadian Journal of Mathematics*, 30(5):997–1015, 1978. doi:10.4153/cjm-1978-085-0.

[83] J. Ian Munro and Venkatesh Raman. Succinct representation of balanced parentheses and static trees. *SIAM Journal on Computing*, 31(3):762–776, 2001. doi:10.1137/S0097539799364092.

[84] Gonzalo Navarro. *Compact Data Structures - A Practical Approach*. Cambridge University Press, 2016. doi:10.1017/cbo9781316588284.

[85] Gonzalo Navarro. Indexing highly repetitive string collections, part I: repetitiveness measures. *ACM Computing Surveys*, 54(2):29:1–29:31, 2022. doi:10.1145/3434399.

[86] Gonzalo Navarro and Luís Manuel. S. Russo. Re-pair achieves high-order entropy. In *Proceedings of Data Compression Conference, DCC 2008*, page 537. IEEE Computer Society, 2008. doi:10.1109/DCC.2008.79.

[87] Craig G. Nevill-Manning and Ian H. Witten. Identifying hierarchical structure in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research*, 7:67–82, 1997. doi:10.1613/jair.374.

[88] Jürg Nievergelt and Edward M. Reingold. Binary search trees of bounded balance. *SIAM Journal on Computing*, 2(1):33–43, 1973. doi:10.1137/0202005.

[89] Carlos Ochoa and Gonzalo Navarro. Repair and all irreducible grammars are upper bounded by high-order empirical entropy. *IEEE Transactions on Information Theory*, 65(5):3160–3164, 2019. doi:10.1109/TIT.2018.2871452.

[90] Christoffer Olsson and Stephan Wagner. Automorphisms of random trees. In *Proceedings of the 33rd International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms, AofA 2022*, volume 225 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.AofA.2022.16.

[91] Richard Otter. The number of trees. *Annals of Mathematics*, 49:583–599, 1948. doi:10.2307/1969046.

[92] Alois Panholzer and Helmut Prodinger. Level of nodes in increasing trees revisited. *Random Structures & Algorithms*, 31(2):203–226, 2007. doi:10.1002/rsa.20161.

[93] Mike Paterson and Mark N. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, 1978. doi:10.1016/0022-0000(78)90043-0.

[94] Eli Plotnik, Marcelo J. Weinberger, and Jacob Ziv. Upper bounds on the probability of sequences emitted by finite-state sources and on the redundancy of the Lempel-Ziv algorithm. *IEEE Transactions on Information Theory*, 38(1):66–72, 1992. doi:10.1109/18.108250.

[95] Nicola Prezza. Optimal rank and select queries on dictionary-compressed text. In *Proceedings of the 30th Annual Symposium on Combinatorial Pattern Matching, CPM 2019*, volume 128 of *LIPIcs*, pages 4:1–4:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CPM.2019.4.

[96] Dimbinaina Ralaivaosaona and Stephan Wagner. Repeated fringe subtrees in random rooted trees. In *Proceedings of the Twelfth Workshop on Analytic Algorithmics and Combinatorics, ANALCO 2015*, pages 78–88. SIAM, 2015. doi:10.1137/1.9781611973761.7.

[97] Dimbinaina Ralaivaosaona and Stephan Wagner. A central limit theorem for additive functionals of increasing trees. *Combinatorics, Probability and Computing*, 28(4):618–637, 2019. doi:10.1007/s00453-019-00622-4.

[98] Rajeev Raman and Srinivasa Rao Satti. Succinct representations of ordinal trees. In *Space-Efficient Data Structures, Streams, and Algorithms - Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday*, volume 8066 of *Lecture Notes in Computer Science*, pages 319–332. Springer, 2013. doi:10.1007/978-3-642-40273-9_20.

[99] Walter Rudin. *Principles of mathematical analysis*. McGraw-Hill Book Company, Inc., New York-Toronto-London, 1953.

[100] Frank Ruskey. Generating linear extensions of posets by transpositions. *Journal of Combinatorial Theory, Series B*, 54(1):77–101, 1992. doi:10.1016/0095-8956(92)90067-8.

[101] Robert Sedgewick and Philippe Flajolet. *An introduction to the analysis of algorithms*. Addison-Wesley-Longman, 1996.

[102] Sebastian Wild. *Dual-pivot quicksort and beyond: Analysis of multiway partitioning and its practical potential*. PhD thesis, Technische Universität Kaiserslautern, 2016.

[103] Sebastian Wild. Quicksort is optimal for many equal keys. In *Proceedings of the Fifteenth Workshop on Analytic Algorithmics and Combinatorics, ANALCO 2018*, pages 8–22. SIAM, 2018. doi:10.1137/1.9781611975062.2.

[104] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987. doi:10.1145/214762.214771.

[105] Jie Zhang, En-Hui Yang, and John C. Kieffer. A universal grammar-based code for lossless compression of binary trees. *IEEE Transactions on Information Theory*, 60(3):1373–1386, 2014. doi:10.1109/TIT.2013.2295392.

[106] Sen Zhang, Zhihui Du, and Jason Tsong-Li Wang. New techniques for mining frequent patterns in unordered trees. *IEEE Transactions on Cybernetics*, 45(6):1113–1125, 2015. doi:10.1109/TCYB.2014.2345579.

[107] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978. doi:10.1109/TIT.1978.1055934.