Universal Enumerative Coding for Tree Models

Alvaro Martín, Gadiel Seroussi, Marcelo J. Weinberger

HP Laboratories
HPL-2011-210

Keyword(s):

universal data compression; enumerative coding; tree models; Markov sources; method of types

Abstract:

Efficient enumerative coding for tree sources is, in general, surprisingly intricate -- a simple uniform encoding of type classes, which is asymptotically optimal in expectation for many classical models such as FSMs, turns out not to be so in this case. We describe an efficiently computable enumerative code that is universal in the family of tree models in the sense that, for a string emitted by an unknown source whose model is supported on a known tree, the expected normalized code length of the encoding approaches the entropy rate of the source with a convergence rate $(K/2)(\log n)/n$, where K is the number of free parameters of the model family. Based on recent results characterizing type classes of context trees, the code consists of the index of the sequence in the tree type class, and an efficient description of the class itself using a non-uniform encoding of selected string counts. The results are extended to a twice-universal setting, where the tree underlying the source model is unknown.

External Posting Date: November 6, 2011 [Fulltext] Approved for External Publication
Internal Posting Date: November 6, 2011 [Fulltext]

© Copyright 2011 Hewlett-Packard Development Company, L.P.

Universal Enumerative Coding for Tree Models*

Álvaro Martín†
Instituto de Computación, Universidad de la República
Montevideo, Uruguay
Email: almartin@fing.edu.uy

Gadiel Seroussi
Universidad de la República, Montevideo, Uruguay
and Hewlett-Packard Laboratories
Palo Alto, CA 94304, USA
Email: gserousi@ieee.org

Marcelo J. Weinberger
Hewlett-Packard Laboratories
Palo Alto, CA 94304, USA
Email: marcelo.weinberger@hp.com

Abstract

Efficient enumerative coding for tree sources is, in general, surprisingly intricate—a simple uniform encoding of type classes, which is asymptotically optimal in expectation for many classical models such as FSMs, turns out not to be so in this case. We describe an efficiently computable enumerative code that is universal in the family of tree models in the sense that, for a string emitted by an unknown source whose model is supported on a known tree, the expected normalized code length of the encoding approaches the entropy rate of the source with a convergence rate $(K/2)(\log n)/n$, where K is the number of free parameters of the model family. Based on recent results characterizing type classes of context trees, the code consists of the index of the sequence in the tree type class, and an efficient description of the class itself using a non-uniform encoding of selected string counts. The results are extended to a twice-universal setting, where the tree underlying the source model is unknown.

*This paper was presented in part at the 2008 International Symposium on Information Theory (ISIT'08), Toronto, Canada, 2008, and part of it was included in the Festschrift in Honor of Jorma Rissanen on the Occasion of his 75th Birthday, 2008.

†Supported by grant PDT - S/C/IF/63/147.

I. INTRODUCTION

In the *method of types* [1], given a parametric model family, the set of sequences of length n over a finite alphabet $\mathcal{A}$ is partitioned into *type classes*, where two sequences belong to the same class if and only if they are assigned the same probability by all models in the family.[1] Since all sequences in a class are equiprobable, the universal probability assignment problem for the given model family reduces to optimally assigning probabilities to type classes.

This reduction is optimally performed by the Normalized Maximum Likelihood (NML) code [3], which can be interpreted as a description of the type, generated by assigning to it a probability proportional to its ML probability, followed by an enumeration of the sequences in the type class. Unfortunately, in general, implementing the NML code is difficult for most popular model families, including the family of tree models studied in this paper (see [4] and references therein for efficient implementation of NML for some specific model families). Other universal methods, based, for example, on the Krichevskii-Trofimov sequential probability assignment [5], are computationally efficient, and also assign the same code length to all the sequences of a given type. They do not, however, provide a separate and identifiable description of the type. In this paper, we are interested in *universal enumerative codes* that possess both qualities: they provide a separate description of the type class of the encoded sequence, and this description can be efficiently computed. By "universal" we mean codes whose normalized average length differs from the entropy rate of the source by $K\frac{\log n}{2n}$ plus lower order terms,[2] where K is the number of free statistical parameters in the family, matching Rissanen's lower bound [6, Theorem 1]. By "efficient computation" we mean one whose encoding time is polynomial in the length of the input sequence, and with code construction time that is also polynomial in the number of free parameters of the family.[3]

For (non-curved) exponential families of probability distributions (see, e.g., [7]) satisfying some mild regularity conditions, most type classes have, to first approximation, the same ML probability [8, Appendix A]. This observation leads to enumerative source codes that are universal (in expectation), and for which uniform coding is used both for the set of type classes and for the set of sequences of each type class. In particular, for *finite–state machine* (FSM) models [9],[4] such a code can be efficiently implemented. Indeed,

[1]Type classes were defined in terms of empirical distributions for memoryless models in [1]. The more general definition of type class used here was introduced in [2, Sec. VII], where extensions of the method of types to wider model families are considered.

[2]Unless specified otherwise, logarithms are to base two.

[3]Our complexity requirement will focus on the description of the type class, as there are known efficient methods to do the enumeration of the class itself for most cases of interest. Notice that, since the number of types in the cases of interest is generally exponential in the number of free parameters of the family, a construction of the NML code relying on the computation of the ML probability of each type would be very inefficient.

[4]Although FSM models are not strictly exponential families (they are curved exponential families), conditioning on a fixed final state s does define an exponential family of probability distributions over the length-n sequences with final state s (see, e.g., [10]). Since all sequences in an FSM type class share the same final state, conditioning the probability of a type class on its final state, say s, amounts to dividing the probability of the type class by the probability of observing state s at time n. This division does not affect the mentioned approximate equiprobability of the ML probability for most type classes.

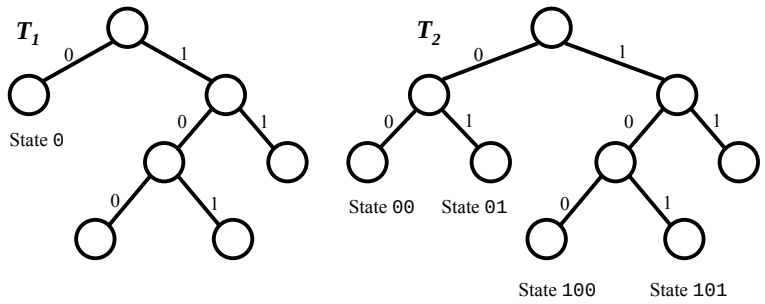

Fig. 1. Tree models over $\mathcal{A} = \{0, 1\}$

for an FSM F with a finite set of states S and alphabet of size α, there are[5] $\Theta\left(n^{|S|(\alpha-1)}\right)$ type classes of sequences of length n [11, attributed to N. Alon]. Thus, a uniform encoding of the type class gives a normalized cost of up to $|S|(\alpha-1)\frac{\log n}{n} + O(\frac{1}{n})$. Moreover, upper bounding the size of a type class by a trivial combinatorial expression, using Stirling's formula, and bounding the expectation as in [12], one obtains, for the type class of a random sequence, that the expected normalized logarithm of the class size is upper-bounded by $\mathcal{H} - |S|(\alpha-1)\frac{\log n}{2n} + O(\frac{1}{n})$, where $\mathcal{H}$ is the entropy rate of the source. Thus, the term subtracted from $\mathcal{H}$ compensates for half the cost of describing the type class, yielding an overall penalty of $|S|(\alpha-1)\frac{\log n}{2n} + O(1/n)$ over $\mathcal{H}$, which is universal, as it matches the bound of [6]. Since the description of the type class is not difficult in this case, and efficient methods exist for enumerating the class, the resulting enumerative code satisfies our requirements.

The question arises: Is a similar technique applicable to useful model families where approximate ML equiprobability of the type classes may not hold? In this paper, we address this question for the popular *tree models* [12], [13], [14], which, in general, do not induce an exponential family of distributions [15] or equiprobable types, and have proven very valuable as modeling tools in data compression and other applications in information theory and statistics (cf. [12], [13], [14], [16], [17]).

A tree model over a finite alphabet $\mathcal{A}$ of size $|\mathcal{A}| = \alpha$ consists of a full α-ary tree[6] T and a set of conditional probability distributions on $\mathcal{A}$, one associated with each leaf of the tree. Each edge of the tree is labeled with a symbol from $\mathcal{A}$ and each node is labeled with the concatenation of the edge labels found on the way from the root to that node. The probability of the next symbol, x_{i+1}, of a sufficiently long string given all the past, x^i, is determined by the conditional probability distribution associated to the unique leaf in the tree whose reversed label is a suffix of x^i, i.e., the leaf obtained by descending from the root, matching the labels of the edges with the symbols in the string, starting from the last symbol and progressing in reverse order. The leaves of the tree are called, thereby, the *states* of the model. In the

[5]We use conventional asymptotic notation: $O(f(n))$ denotes a function $g(n)$ such that $|g(n)| \leq c|f(n)|$ for a positive constant c and sufficiently large n, $\Theta(f(n))$ a function $g(n)$ such that $g(n) = O(f(n))$ and $f(n) = O(g(n))$, and $o(f(n))$ a function $g(n)$ such that $\lim_{n\to\infty} g(n)/f(n) = 0$.

[6]We say that an α-ary tree T is *full* if every internal node of T has exactly α children.

binary tree T_1 of Figure 1, for example, all strings ending with 0 select State 0, while in tree T_2, strings ending with 00 select State 00 and strings ending with 10 select State 01.

The number of type classes for a tree model with set of states S_T grows polynomially as n^k, but the exponent k may be larger than $(\alpha-1)|S_T|$, which would be the exponent in the FSM case. The asymptotic number of type classes for a tree model is fully characterized in [18], generalizing the corresponding result for FSMs. A simple example that illustrates the discrepancy with the FSM case is easy to construct: Consider the binary trees T_1 and T_2 of Figure 1. Since the number of occurrences of symbol 1 in states 00 and 01 of T_2 are determined, respectively, by the number of occurrences of states 100 and 101, it is not difficult to see that each type class of T_1 is partitioned into up to a constant number of type classes in T_2. Now, the tree T_2 has the "FSM property" (in the sense that the occurrence of a symbol in a state defines the next state), so the number of type classes, by the above discussion on FSMs, is $\Theta(n^5)$ for both trees, even though T_1 has four states. Since, as noted, the partitions are essentially the same, lower-bounding the average code length appropriately, we can readily see that an enumerative scheme using a uniform code for the type classes would not exploit the reduction in the number of model parameters of T_1 with respect to T_2. Notice, however, that a type class with significantly different conditional empirical distributions for states 00 and 01 of T_2 will have small probability under the model T_1 for *any* choice of model parameters, which suggests that the savings in code length might be recovered with a *non-uniform* code for the type classes.

In this paper, we construct such non-uniform code, leading to an efficient enumerative coding scheme which is universal (in expectation) for the family of tree models. Furthermore, in the twice–universal setting, in which a tree is not given and optimality is rather required for *any* possible tree, we show that, by suitably estimating a tree from the data, the sequences in the aforementioned "atypical" type classes for each given tree would in fact estimate a different tree. These type classes can thus be discarded from the coding space, leading to a twice–universal enumerative code in the family of tree models.

Our implementation of the second part of the enumerative code, namely, the index of x^n in its type class, will be based on the enumeration of tree type classes from [18]. As for the first part, namely, the description of the type class, a key building block in our scheme will be a collection of codes for encoding counts of occurrences of certain patterns within the input sequence. In the example of Figure 1, given the empirical conditional distribution, $\hat{p}$, in state 0 of T_1, and the number of occurrences, n_s, of the pattern 00, we can estimate the number of occurrences, $n_s^{(a)}$, of symbol a in state 00 of T_2 as $\hat{n}_s^{(a)} = n_s\hat{p}(a)$. If n_s and $\hat{p}$ have already been described to the decoder, we can then encode the difference $n_s^{(a)} - \hat{n}_s^{(a)}$ by assigning high probability to small absolute differences. This observation will be generalized to encode, efficiently, a collection of pattern counts that uniquely determines the type class of a sequence.

The rest of the paper is organized as follows. In Section II we introduce our notation and formal setting, and review some results from [16] and [18]. In Section III we present a universal enumerative code for tree models, for which we introduce variable length codes for the encoding of pattern counts, following

the observation above. These codes, which are based on Golomb codes [19] and dubbed *string count codes* (*SCC*), will be used for the encoding of type classes and will also help us obtain a precise asymptotic upper bound on the expected size of the type class of a sequence with respect to a given tree. Finally, in Section IV we present two approaches for the twice-universal setting. The first approach is a standard plug-in scheme where the tree is first estimated, and then the previously derived universal enumerative code is applied as-is, using the estimated tree in lieu of a given one. In the second approach, we take advantage of the observation above that for any given tree, there will be sequences that are "atypical" for the tree, and will not estimate it regardless of the model parameters. Thus, the enumerative code is significantly simplified in the twice-universal setting by excluding such sequences from the coding space for the estimated tree.

II. Background and Preliminaries

A. Tree models

Let $\mathcal{A}$ be an alphabet of $\alpha \geq 2$ symbols and let $\mathcal{A}^*$, $\mathcal{A}^+$, and $\mathcal{A}^n$ denote, respectively, the set of finite strings, positive-length strings, and length-n strings over $\mathcal{A}$. We denote by u_j^k the string $u_j u_{j+1} \dots u_k$ over $\mathcal{A}$, with $u_j^k = \lambda$, the empty string, when $j > k$. We omit the subscript when $j = 1$. For a string $u = u^k$, we let $\overline{u} = u_k u_{k-1} \dots u_1$ denote its reverse, $|u| = k$ its length, and we write $\mathrm{head}(u) = u_1$, and $\mathrm{tail}(u) = u_2^k$; $|S|$ also denotes the cardinality of a set S. Concatenation of u and v is denoted uv, and $u \preceq v$ (resp. $u \prec v$) denotes the prefix (resp. proper prefix) relation.

Our models will be based on directed full α-ary trees (or simply, *trees*), in which each of the α edges departing from an internal node is labeled with a different symbol from $\mathcal{A}$, and each node is labeled with the string formed by concatenating the edge labels on the path from the root (labeled by λ) to the node. We identify a tree T with its set of nodes, and each node with its label, e.g., $u \in T$ indicates that there is a node of T labeled u. The leaves of T are called *states*. The set of states is denoted S_T and we let $\mathcal{I}(T) = T \setminus S_T$ denote the set of internal nodes of T. For a set of nodes of a tree, W, we let $\mathcal{P}(W)$ denote the set or parents of nodes in W, i.e.,

$$\mathcal{P}(W) = \{u \in \mathcal{A}^* : ua \in W \text{ for some } a \in \mathcal{A}\}. \tag{1}$$

Although the definition (1) can be applied to an arbitrary set of strings W, we will always regard W as a subset of some tree.

For a sufficiently long sequence x^n, we refer to the (unique) prefix of $\overline{x^n}$ in S_T, denoted $\sigma(x^n)$, as the state *selected* by x^n (the dependence of σ on T will be assumed from the context). For the purpose of selecting states, we assume that x^n is preceded by an arbitrary *fixed* semi-infinite string $x_{-\infty}^0$. This convention uniquely determines, for any given tree, an *initial state* s_0 "selected" by λ, and guarantees that any (short) sequence selects a state. Thus, x^n uniquely determines a *state sequence* $s_0, s_1, \dots, s_n$, with $s_i = \sigma(x^i)$, $0 \leq i \leq n$. We refer to s_n as the *final state* of x^n with respect to T and we say that the symbol

x_{i+1} *occurs in state* s_i, $0 \le i < n$. The notion of occurrence is extended to arbitrary strings, namely, if for $u = u^k$ and some index i, $0 \le i < n$, we have $x_{i-k+1}^i = \overline{u}$, we say that x_{i+1} *occurs in context* u in x^n (notice that, for consistency with our state labeling convention, the reverse of u is matched with x^n).

Trees do not necessarily define a *next-state* function. In the tree T_1 of Figure 1, for example, the occurrence of symbol 1 in state 0 does not determine whether the next state will be 100 or 101. When a next state function exists for a tree T we say, concisely, that T *is FSM*. The following theorem characterizes FSM trees.

Theorem 1 ([16, Theorem 2]): A tree T is FSM if and only if every suffix of a state of T is in T.

A tree T' is an *extension* of T if $T \subseteq T'$. If a state s of T is an internal node of an extension T', we say that T' extends s. The tree T_F obtained from T by adding all the suffixes of states of T is called its *FSM closure*; T_F is the minimal FSM extension of T [16]. In Figure 1, T_2 is the FSM closure of T_1.

A *model parameter* for a tree T, denoted p_T, is a set of $|S_T|$ conditional probability distributions over $\mathcal{A}$, one associated to each state of T. The tree T and the model parameter p_T define a *tree model*, denoted by $\langle T, p_T \rangle$, which in turn defines a *probability assignment* [20] $\mathrm{P}_{\langle T,p_T\rangle}(\cdot)$ given by

$$\mathrm{P}_{\langle T,p_T\rangle}(\lambda) = 1; \quad \mathrm{P}_{\langle T,p_T\rangle}(x^n) = \prod_{i=1}^{n} p_T(x_i|s_{i-1}), \;\; n \ge 1\,. \tag{2}$$

For each $n \ge 0$, the assignment (2) determines a probability distribution on $\mathcal{A}^n$ and, thus, $\langle T, p_T \rangle$ completely defines a stochastic process or *source*. We use the notation $\mathrm{P}_{\langle T,p_T\rangle}\{\cdot\}$ to refer to the probability of an event that depends on a random sequence $X^n \in \mathcal{A}^n$ drawn with probability $\mathrm{P}_{\langle T,p_T\rangle}(\cdot)$, where n will be clear from the context. For example, we write $\mathrm{P}_{\langle T,p_T\rangle}\{f(X^n) = 0\}$, for some function f, to denote the probability $\sum_{x^n : f(x^n)=0} \mathrm{P}_{\langle T,p_T\rangle}(x^n)$.

Different trees and sets of conditional probabilities can generate the probability assignment (2); a tree model $\langle T, p_T \rangle$ is *minimal* if for every internal node u of T there exist states uv and uw such that $p_T(\cdot|uv) \not\equiv p_T(\cdot|uw)$. In the sequel, when there is no ambiguity, we will loosely use the symbol T to refer both to a tree model and to its underlying tree. For example, we will simplify the notation $\mathrm{P}_{\langle T,p_T\rangle}\{\cdot\}$ to $\mathrm{P}_T\{\cdot\}$. All expectations $\mathrm{E}[\,\cdot\,]$ will be with respect to $\mathrm{P}_{\langle T,p_T\rangle}\{\cdot\}$, where $\langle T, p_T \rangle$ will be clear from the context.

B. Tree type classes and enumerative coding

For a sequence x^n, a string u, and a symbol $a \in \mathcal{A}$, define

$$n_u^{(a)}(x^n) = |\{\, i \,:\, 0 \le i < n,\; x_{i-|u|+1}^i = \bar{u},\; x_{i+1} = a \,\}|\,, \tag{3}$$

namely, the number of occurrences of a in context u (or, if $u \in S_T$, in state u) in x^n. Define also

$$n_u(x^n) = \sum_{a \in \mathcal{A}} n_u^{(a)}(x^n)\,. \tag{4}$$

We omit the dependence of counts on x^n when clear from the context. Notice that, by (3), we have

$$n_u^{(a)} = \sum_{b \in \mathcal{A}} n_{ub}^{(a)} \tag{5}$$

and, summing (5) over all $a \in \mathcal{A}$, by (4), we obtain

$$n_u = \sum_{b \in \mathcal{A}} n_{ub} \,. \tag{6}$$

Furthermore, denoting by $\mathbf{i}(w)$ and $\mathbf{f}(w)$ the indicator functions of the predicates $\overline{w} = x_{-|w|+1}^{0}$ and $\overline{w} = x_{n-|w|+1}^{n}$, respectively, we have

$$n_{au} + \mathbf{f}(au) = n_u^{(a)} + \mathbf{i}(au) \,, \tag{7}$$

for every string u and every $a \in \mathcal{A}$. To simplify expressions, we will use a generic constant δ to account for border adjustments due to terms of the form $\mathbf{i}(w)$ and $\mathbf{f}(w)$. In coding situations these terms will be known to the decoder, and in any case border effects will have no bearing on the asymptotic results.

From (2), using a simple algebraic argument, it follows that for sequences x^n and y^n, we have

$$\mathrm{P}_{\langle T, p_T \rangle}(x^n) = \mathrm{P}_{\langle T, p_T \rangle}(y^n)$$

for all choices of the model parameter p_T, if and only if $n_s^{(a)}(x^n) = n_s^{(a)}(y^n)$ for all $s \in S_T$, $a \in \mathcal{A}$. Thus, in the case of tree models (as in other cases of interest), the notion of type defined in probabilistic terms in Section I admits a combinatorial characterization. For a tree T, and a sequence x^n, we denote by $\mathcal{K}(T, x^n)$ the collection of counts $\{\, n_s^{(a)} \,\}_{s \in S_T,\, a \in \mathcal{A}}$. The *type class* of x^n with respect to T can then be defined as

$$\mathcal{T}(T, x^n) = \{\, y^n \in \mathcal{A}^n \,:\, \mathcal{K}(T, y^n) = \mathcal{K}(T, x^n) \,\} \,.$$

An encoding of x^n with an *enumerative (source) code* for T is comprised of two parts: a description of $\mathcal{K}(T, x^n)$, and an index of x^n within $\mathcal{T}(T, x^n)$. By using a trivial bound

$$|\mathcal{T}(T, x^n)| \leq \prod_{s \in S_T} \frac{n_s!}{\prod_{a \in \mathcal{A}} n_s^{(a)}!} \,,$$

Stirling's formula, and applying bounds on expectations from [12], we obtain the following lemma, which bounds the length of the second part of the enumerative code.

Lemma 1: Let T be a tree model with all conditional probabilities nonzero, and let $\mathcal{H}$ be the entropy rate of the corresponding source. Then,

$$\frac{1}{n} \mathrm{E}\left[\, \log |\mathcal{T}(T, x^n)| \,\right] \leq \mathcal{H} - |S_T|(\alpha - 1) \frac{\log n}{2n} + O\left(\frac{1}{n}\right) . \tag{8}$$

The following theorem states that we can efficiently implement the second part of the enumerative code, which involves computing an enumeration of $\mathcal{T}(T, x^n)$, i.e., a one-to-one mapping f from $\mathcal{T}(T, x^n)$ to the set of indexes $\{\, 0, 1, \ldots, |\mathcal{T}(T, x^n)| - 1 \,\}$, and its inverse, f^{-1}.

Theorem 2: Given $\mathcal{K}(T, x^n)$, an enumeration of $\mathcal{T}(T, x^n)$ and its inverse can be computed in time that is polynomial in n and in $|T|$.

Proof: It is shown in [18, Theorem 3] that, given the so-called *pseudo-state transition matrix* [18] of x^n with respect to T, an enumeration f of $\mathcal{T}(T, x^n)$ and its inverse, f^{-1}, can be computed in time that is polynomial in n. Given $\mathcal{K}(T, x^n)$, the construction of the pseudo-state transition matrix, in turn, can be done in time polynomial in $|T|$, as it follows from [18, Lemma 14]. ■

C. Minimal canonical extension

We say that a state $s \in S_T$ is *forgetful* if $as \in \mathcal{I}(T)$ for *all* $a \in \mathcal{A}$. In a forgetful state, a next-state transition cannot be determined for *any* occurring symbol $a \in \mathcal{A}$. A tree with no forgetful states is called *canonical*. It can be shown [18, Lemma 2] that, by sequentially extending forgetful states until no such state is left, a tree T is brought to a unique *minimal canonical extension*, which will be denoted $T_{\mathbf{c}}$. Any canonical extension of T is an extension of $T_{\mathbf{c}}$. In a sense, forgetful states are the farthest away from satisfying the FSM property, and $T_{\mathbf{c}}$ brings a tree T "closer" to its FSM closure T_F. Thus, we have the following relation among T, $T_{\mathbf{c}}$, and T_F.

Lemma 2: The trees T, $T_{\mathbf{c}}$, and T_F satisfy $T \subseteq T_{\mathbf{c}} \subseteq T_F$, and they are all of the same depth.

Proof: Since $T_{\mathbf{c}}$ is an extension of T, we have $T \subseteq T_{\mathbf{c}}$. T_F is also an extension of T and it is clearly canonical, as it defines a next-state function. Since any canonical extension of T is an extension of $T_{\mathbf{c}}$, we must have $T_{\mathbf{c}} \subseteq T_F$. Finally, since T_F is obtained from T by adding all the suffixes of states of T, T_F and T are the same depth. ■

We denote by $\sigma_{\mathbf{c}}(u)$ the state selected by u in $T_{\mathbf{c}}$. The following lemma follows immediately from [18, Lemma 8] and [18, Corollary 2].

Lemma 3: $\mathcal{K}(T_{\mathbf{c}}, x^n)$ can be reconstructed from a description of $\mathcal{K}(T, x^n)$ and $\sigma_{\mathbf{c}}(x^n)$. This computation takes a number of operations on registers of length $O(\log n)$ bits that is polynomial in $|T|$.

D. The number of tree type classes

We define the *state transition support graph* $G_T{=}(V_T, E_T)$ of a tree T, with vertex set $V_T{=}S_T$, and edge set E_T comprising all state pairs (s, t) such that some sequence x^n causes a direct transition from s to t in T. By the definition of state selection in trees, the symbol x_{i+1} causes such a transition if and only if $\text{head}(t) = x_{i+1}$ and the reverse of both s and $\text{tail}(t)$ are prefixes of x^i. Thus, we have

$$E_T = \{\, (s,t) \,:\, s \preceq \text{tail}(t) \text{ or } \text{tail}(t) \prec s \,\} \,. \tag{9}$$

The following theorem establishes the asymptotic number of type classes induced by a tree T on sequences of length n, in terms of the state transition support graph of $T_{\mathbf{c}}$.

Theorem 3 ([18, Theorem 4]): The number of type classes induced by T on sequences of length n is

$$\mathcal{N}_T = \Theta\left(n^{|E_{T_{\mathbf{c}}}|-|V_{T_{\mathbf{c}}}|}\right) \,. \tag{10}$$

When T is FSM, T is also canonical, and there are exactly α edges departing from each state in T. Thus, in this case, the exponent in (10) is $|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}| = |S_T|(\alpha - 1)$. Thus, $(\alpha{-}1)|S_T| \log n + O(1)$ bits suffice to describe $\mathcal{T}(T, x^n)$ when T is FSM and, therefore, by (8), an enumerative code for an FSM tree T, based on uniform coding of the type class, is universal (in expectation). Moreover, such an enumerative code can be efficiently implemented, as it follows from Theorem 2 and Lemma 4 below. We denote by $\mathcal{K}_b(T, x^n)$, $b{\in}\mathcal{A}$, the collection of $|S_T|(\alpha - 1)$ counts

$$\mathcal{K}_b(T, x^n) = \{n_s^{(a)}\}_{s \in S_T,\, a \in \mathcal{A}\setminus\{b\}} \,. \tag{11}$$

Lemma 4: If a tree T is FSM, then for any $b \in \mathcal{A}$, $\mathcal{K}_b(T, x^n)$ and the final state, $\sigma(x^n)$, of x^n in T completely determine $\mathcal{K}(T, x^n)$. The computation of $\mathcal{K}(T, x^n)$ takes a number of operations on registers of length $O(\log n)$ bits that is polynomial in $|T|$.

Lemma 4 follows by applying [18, Lemma 15(ii)] to the state transition support graph of a tree that is FSM. For a general tree model, however, $\mathcal{K}_b(T, x^n)$ and $\sigma(x^n)$ are insufficient to determine $\mathcal{K}(T, x^n)$. The exponent $|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|$ in (10) may be larger than $|S_T|(\alpha - 1)$ and, as discussed in Section I, an enumerative code with a uniform encoding of type classes may be suboptimal.

III. Universal enumerative coding

In this section, we present an efficiently computable description of the type class $\mathcal{T}(T, x^n)$ (or, equivalently, the counts $\mathcal{K}(T, x^n)$). By Theorem 2, the index of a sequence within its type class can be computed and uniformly encoded, efficiently, using the combinatorial characterization of the tree type class in [18]. These descriptions of $\mathcal{K}(T, x^n)$ and the index of x^n within its type class, together, yield an enumerative code for tree models. We also analyze the expected code length of this code showing that it is universal in the family of tree models. For this purpose, we study the expected length of the the proposed description of $\mathcal{K}(T, x^n)$ (code length of the first part of the enumerative code), and the asymptotic behavior of $\mathrm{E}\,[\log|\mathcal{T}(T, X^n)|\,]$ (code length of the second part of the enumerative code). We assume, throughout, that the tree is not trivial, i.e., $|S_T| > 1$.

A. Outline

Our encoding of type classes will consist of two parts: a description of $\mathcal{K}_b(T, x^n)$ as defined in (11) for some $b \in \mathcal{A}$ (which is generally insufficient to characterize the type class), and a set of additional counts that completes the description of $\mathcal{K}(T, x^n)$. These additional counts can be efficiently described, as stated in the following lemma.

Lemma 5: Let T be a tree model with all conditional probabilities different from zero and let X^n be a random sequence emitted by the corresponding source. Then, there exists a code for describing $\mathcal{K}(T, x^n)$ whose code length, $L(X^n)$, satisfies

$$\mathrm{E}\,[\,|L(X^n)|\,] \le \frac{1}{2}(\alpha - 1)|S_T| \log n + \frac{1}{2}\left(|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|\right) \log n + O(1)\,. \tag{12}$$

Moreover, such a code can be obtained by encoding $\mathcal{K}_b(T, x^n)$, using $|S_T|(\alpha-1)$ counts of $\log n$ bits[7] each, the final state of x^n in $T_\mathbf{c}$, using a constant number of bits, and an additional $|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}| - |S_T|(\alpha - 1)$ counts, requiring on average $\frac{1}{2}\log n + O(1)$ bits of description each. The overall encoding of $\mathcal{K}(T, x^n)$ requires a number of operations on registers of length $O(\log n)$ that is polynomial in $|T|$ and in n.

The crux of the proof of Lemma 5 will be to find a minimum set of counts, and the order in which they are described, that suffice to determine $\mathcal{K}(T, x^n)$, and, at the same time, can be described economically.

[7]To simplify discussions, we will sometimes ignore integer constraints on code lengths, referring, for example, to $\log n$ bits instead of the more precise "at most $\lceil \log n \rceil$ bits." This loose convention will be immaterial to the main asymptotic results of the paper.

In Subsection III-B we present our enumerative code by defining, for the first part, algorithmically, the set of counts to be encoded, the order in which they are encoded, and how $\mathcal{K}(T, x^n)$ can be reconstructed by the decoder. The encoding of the counts themselves is defined in Subsection III-C, where we introduce *string count codes (SCC)*. As mentioned, the second part of the enumerative code will be based on the enumeration of the tree type class in [18].

SCC will also serve as a tool, in a coding argument, to analyze the asymptotic behavior of the expected type class size in Subsection III-D. When T is FSM, the bound (8) on the expected type class size leads readily, by means of Lemma 4, to the optimality of enumerative coding where type classes are encoded uniformly. For general trees, however, describing the type class requires, by Lemma 5, an average of $\frac{1}{2}\left(|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}| - (\alpha-1)|S_T|\right)\log n$ additional bits with respect to the $(\alpha-1)|S_T|\log n + O(1)$ bits that, by Lemma 4, suffice in the FSM case. Thus, we need a tighter upper-bound on $\mathrm{E}\left[\log|\mathcal{T}(T, X^n)|\right]$ to offset the length increase of the type class description part. In Subsection III-D, we prove the bound in Theorem 4 below, which satisfies our requirements.

Theorem 4: Let T be a tree model with all conditional probabilities different from zero, $\mathcal{H}$ the entropy rate of the corresponding source, and $G_{T_\mathbf{c}}{=}(V_{T_\mathbf{c}}, E_{T_\mathbf{c}})$ the state transition support graph of $T_\mathbf{c}$. Then, for a random sequence X^n emitted by this source,

$$\frac{1}{n}\mathrm{E}\left[\log|\mathcal{T}(T, X^n)|\right] \leq \mathcal{H} - \frac{|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|}{2n}\log n + O\left(\frac{1}{n}\right).$$

Notice that, again, when T is FSM, we have $|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}| = |S_T|(\alpha-1)$, bringing Theorem 4 in agreement with Lemma 1. Theorem 4 complements, by providing an asymptotic interpretation, the exact combinatorial characterization of the tree type class in [18]. While the combinatorial characterization is instrumental in implementing the enumerative code, the asymptotic result helps us estimate the average code length.

We prove Lemma 5 in Subsection III-E. Together with Theorems 2 and 4, these results will show that the proposed enumerative code is universal and efficiently implementable, as summarized in Theorem 5 below, which is our main result. This theorem is also proved in Subsection III-E.

Theorem 5: Let T be a tree model with all conditional probabilities different from zero and let $\mathcal{H}$ be the entropy rate of the corresponding source. Then, there exists an enumerative code for T, which can be efficiently implemented, and whose code length, $L(X^n)$, for a random sequence X^n emitted by this source, satisfies

$$\mathrm{E}\left[\frac{L(X^n)}{n}\right] \leq \mathcal{H} + \frac{|S_T|(\alpha-1)\log n}{2n} + O\left(\frac{1}{n}\right). \tag{13}$$

B. Algorithmic description of the enumerative code

Let h and d denote, respectively, the minimal and maximal depth of leaves in T. For $h \leq k \leq d$, let $T^{[k]}$ denote the truncation of T to depth k, and $T_\mathbf{c}^{[k]}$ the canonical extension of $T^{[k]}$. Notice that, by Lemma 2, $T_\mathbf{c}^{[k]}$ has depth k. We denote by $S_\mathbf{c}^{[k]}$ the set of states of $T_\mathbf{c}^{[k]}$, and by $\sigma_\mathbf{c}^{[k]}(u)$ the state selected by u in $T_\mathbf{c}^{[k]}$.

Algorithm EncodeTypeClass(T, x^n)

1. Choose $b \in \mathcal{A}$; encode $\mathcal{K}_b(T, x^n)$ and the final state of x^n in $T_{\mathbf{c}}$.
   /* By Lemma 4 and the fact that $T^{[h+1]}$ is FSM,
   this completely describes $\mathcal{K}(T^{[h+1]}, x^n)$. */
2. Set $h = \min\{|s| : s \in S_T\}$ and $d = \max\{|s| : s \in S_T\}$.
3. For $k = h + 2$ to d
4. Encode $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ given $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$.

Fig. 2. Encoding of $\mathcal{K}(T, x^n)$

To encode the type class of x^n with respect to T we will describe $\mathcal{K}(T_{\mathbf{c}}, x^n)$, which, since $T_{\mathbf{c}}$ is an extension of T, suffices to reconstruct $\mathcal{K}(T, x^n)$. This description, in turn, will be done by truncation levels, starting from a description of $\mathcal{K}(T_{\mathbf{c}}^{[h+1]}, x^n)$, and progressively describing each count set $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$, for $h + 2 \le k \le d$, based on the previous description of the count set $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$.

Algorithm *EncodeTypeClass*, shown in Figure 2, lists the main steps in the proposed encoding of $\mathcal{K}(T, x^n)$. The algorithm starts by encoding, uniformly, the counts in $\mathcal{K}_b(T, x^n)$ for an arbitrary $b \in \mathcal{A}$ with $\log n$ bits per count, and the final state of x^n in $T_{\mathbf{c}}$, using a constant number of bits. It then iterates to describe, incrementally, each count set $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ given a previously described set $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$, for $h + 2 \le k \le d$. Notice that since $T^{[h]}$ is a full balanced tree, $T^{[h+1]}$ is FSM, and, hence, $T_{\mathbf{c}}^{[h+1]} = T^{[h+1]}$. By Lemma 4, $\mathcal{K}(T^{[h+1]}, x^n)$ is completely determined by $\mathcal{K}_b(T, x^n)$ (which, clearly, describes also $\mathcal{K}_b(T^{[h+1]}, x^n)$) and the given final state. Thus, a decoder can reconstruct $\mathcal{K}(T_{\mathbf{c}}^{[h+1]}, x^n)$ from the information provided in Step 1 of EncodeTypeClass, and can recover $\mathcal{K}(T_{\mathbf{c}}^{[d]}, x^n)$ from the information encoded in the loop of Steps 3–4 of the algorithm. As mentioned, since $T_{\mathbf{c}}^{[d]}$ is an extension of T, this is sufficient to reconstruct $\mathcal{K}(T, x^n)$.

Clearly, the crucial step in EncodeTypeClass is the encoding of the refinement of counters from $T_{\mathbf{c}}^{[k-1]}$ to $T_{\mathbf{c}}^{[k]}$ in Step 4. For $u, v \in \mathcal{A}^*$, we denote by $N_{u,v}$ the number of times a transition from context u to context v occurs in x^n (i.e., the number of indexes i, $0 < i < n$, such that x_i occurs in context u and x_{i+1} occurs in context v). In particular, when u and v are states of a tree, $N_{u,v}$ denotes the number of times state v is selected immediately after state u. Notice that, for a state t, we have

$$n_t = \sum_{s \in S_T} N_{t,s} = \sum_{s \in S_T} N_{s,t} + \delta \,. \tag{14}$$

Our implementation of Step 4 will amount to describing all state transition counts $N_{s,t}$, with $s, t \in S_{\mathbf{c}}^{[k]}$. This set of counts is sufficient to determine $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ since $n_s^{(a)} = \sum_{au \in S_{\mathbf{c}}^{[k]}} N_{s,au}$. However, not all the counters in the set will be explicitly described, since some will be derivable from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$ and earlier portions of the description of $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$. All explicit encodings will be for counts $N_{s,t}$ with $s, t \in S_{\mathbf{c}}^{[k]}$ satisfying $s \preceq \mathrm{tail}(t)$ (which, by (9), is a subset of all state transitions), for which we will

make use of the following lemma.

Lemma 6: Let s and t be states of a tree such that $s \preceq \mathrm{tail}(t)$. Then, $N_{s,t} = n_t + \delta$.

Proof: Notice that s is the only source of state transitions into t and apply (14). ■

The proposed encoding of $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ given $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$ will rely on relations between the consecutive truncations, $T^{[k-1]}$ and $T^{[k]}$, of T, and their respective canonical extensions, $T_{\mathbf{c}}^{[k-1]}$ and $T_{\mathbf{c}}^{[k]}$. We investigate these relations next. We define the *truncation increment*,

$$\Delta T^{[k]} = T^{[k]} \setminus T^{[k-1]}\,, \quad h < k \leq d\,, \tag{15}$$

namely the set of nodes of $T^{[k]}$ at level k. The following lemma is an immediate consequence of the definition of $T^{[k]}$. We recall that $\mathcal{I}(T)$ denotes the set of internal nodes of a tree T, and $\mathcal{P}(W)$, defined in (1), denotes the set of parents of a set of nodes W.

Lemma 7: Let $h < k \leq d$. Then, we have

$$\Delta T^{[k]} = S_{T^{[k]}} \setminus S_{T^{[k-1]}}\,, \quad \text{and} \quad \mathcal{P}\left(\Delta T^{[k]}\right) = S_{T^{[k-1]}} \setminus S_{T^{[k]}} = \{r \in \mathcal{I}\left(T^{[k]}\right) : |r| = k-1\}\,.$$

In analogy to (15), we define the *canonical truncation increment*,

$$\Delta T_{\mathbf{c}}^{[k]} = T_{\mathbf{c}}^{[k]} \setminus T_{\mathbf{c}}^{[k-1]}\,, \quad h < k \leq d\,. \tag{16}$$

Notice that, in general, $T_{\mathbf{c}}^{[k]}$ is different from $(T_{\mathbf{c}})^{[k]}$ and, therefore, $\Delta T_{\mathbf{c}}^{[k]}$ is not necessarily the set of nodes of $T_{\mathbf{c}}$ at level k. It is true, however, that $T_{\mathbf{c}}^{[k]}$ is an extension of $T_{\mathbf{c}}^{[k-1]}$, as stated in the following lemma, whose proof is deferred to Appendix A.

Lemma 8: Let $h < k \leq d$. Then, $T_{\mathbf{c}}^{[k]}$ is an extension of $T_{\mathbf{c}}^{[k-1]}$.

By Lemma 8, $T_{\mathbf{c}}^{[k]}$ is obtained from $T_{\mathbf{c}}^{[k-1]}$ by adding the nodes in $\Delta T_{\mathbf{c}}^{[k]}$. We distinguish between nodes in $\Delta T_{\mathbf{c}}^{[k]}$ that are added to $T_{\mathbf{c}}^{[k-1]}$ for they belong to $T^{[k]}$ (but not to $T^{[k-1]}$), and nodes in $\Delta T_{\mathbf{c}}^{[k]}$ that arise in $T_{\mathbf{c}}^{[k]}$ in order to take $T^{[k]}$ to canonical form. The latter set, called the *canonization increment*, is given by

$$\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]} = \Delta T_{\mathbf{c}}^{[k]} \setminus T^{[k]}\,, \quad h < k \leq d\,. \tag{17}$$

The following lemma will be instrumental in identifying an appropriate set of counts to describe $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ given $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$.

Lemma 9: For $h < k \leq d$, $a \in \mathcal{A}$, and $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, we have $ar \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$.

Proof: Since $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, by (17) and (16), r is the parent of a node in $T_{\mathbf{c}}^{[k]} \setminus T^{[k]}$ and, therefore, by the definition of canonical tree, r is a forgetful state of some tree T' built in the process of making $T^{[k]}$ canonical (recall the process for making a tree canonical from Subsection II-C). Thus, by the definition of forgetful state, ar is an internal node of some tree T' satisfying $T^{[k]} \subseteq T' \subseteq T_{\mathbf{c}}^{[k]}$. ■

The following lemma establishes some relations among the objects in the foregoing definitions. The proof is deferred to Appendix A.

Lemma 10: Let $h < k \leq d$. Then,

Procedure RefineTypeClass

1. For each $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ taken in non-decreasing order of length $|r|$:
2. If $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$
   /* Since $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, by (19), this implies $r \in S_{\mathbf{c}}^{[k-1]}$. */
3. Take $d \in \mathcal{A}$ such that $dr \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$.
   /* Such d must exist (see proof of Lemma 12). */
4. Use $\mathtt{P}(r, c)$ to describe counts $N_{s,t}$
   for every $s \in S_{\mathbf{c}}^{[k]}(r)$, $c \in \mathcal{A}_d$, and $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \mathrm{head}(t)$.
5. [Let $N_{s,ds} = n_s^{(d)} = n_s - \sum_{c \neq d} n_s^{(c)}$, for all $s \in S_{\mathbf{c}}^{[k]}(r)$.]
   /* We have $ds \in S_{\mathbf{c}}^{[k]}$ and n_s is known (see proof of Lemma 12).
      The last equality follows from (4). */
6. else, If $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$
7. For each $c \in \mathcal{A}$ such that $cr \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$,
8. [For each $s \in S_{\mathbf{c}}^{[k]}(r)$, $s' = \sigma_{\mathbf{c}}^{[k]}(\overline{cs})$ is well-defined, thus $N_{s,s'} = n_s^{(c)}$.]
   /* Since $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$, the child s of r belongs to $T_{\mathbf{c}}^{[k-1]}$,
      therefore $n_s^{(c)}$ is known from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. */
9. For each $c \in \mathcal{A}$ such that $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$,
10. Use $\mathtt{P}(r, c)$ to describe counts $N_{s,t}$
    for every $s \in S_{\mathbf{c}}^{[k]}(r)$, and every $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \mathrm{head}(t)$.
11. else, /* By Lemma 11, $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$. */
12. For each $s \in S_{\mathbf{c}}^{[k]}(r)$, /* By Lemma 7, $s \in S_{T^{[k]}}$ and $|s| = k$. */
13. [For each $a \in \mathcal{A}_b$, let $s' = \sigma_{\mathbf{c}}^{[k]}(\overline{as})$ and therefore $N_{s,s'} = n_s^{(a)}$.]
    /* Since $s \in S_{T^{[k]}}$, $n_s^{(a)}$ is known from $\mathcal{K}_b(T, x^n)$. */
14. [For $s' = \sigma_{\mathbf{c}}^{[k]}(\overline{bs})$, take $N_{s,s'} = n_s^{(b)} = n_s - \sum_{a \neq b} n_s^{(a)}$.]
    /* n_s is known (see proof of Lemma 12).
       The last equality follows from (4). */

Fig. 3. Encoding and decoding of $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$

(i) $T_{\mathbf{c}}^{[k]}$ extends states of $T_{\mathbf{c}}^{[k-1]}$ by at most one level, i.e., $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq T_{\mathbf{c}}^{[k-1]}$.

(ii) Given $r \in \mathcal{A}^*$ and $a \in \mathcal{A}$, we have $r \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ if and only if $ra \in \Delta T_{\mathbf{c}}^{[k]}$.

(iii) The sets $\Delta T_{\mathbf{c}}^{[k]}$ and $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ satisfy

$$\Delta T_{\mathbf{c}}^{[k]} = S_{\mathbf{c}}^{[k]} \setminus S_{\mathbf{c}}^{[k-1]}, \tag{18}$$

$$\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) = S_{\mathbf{c}}^{[k-1]} \setminus S_{\mathbf{c}}^{[k]} = S_{\mathbf{c}}^{[k-1]} \cap \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) = \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right). \tag{19}$$

We now present the implementation of Step 4 of EncodeTypeClass, which is shown as Procedure *RefineTypeClass* in Figure 3. Decoding steps are shown in brackets, to verify the losslessness of the code. In RefineTypeClass, b is the same symbol as in Step 1 of EncodeTypeClass. For $d \in \mathcal{A}$, we let $\mathcal{A}_d = \mathcal{A} \setminus \{d\}$. For $h \leq k \leq d$ and $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, we denote by $S_{\mathbf{c}}^{[k]}(r)$ the subset of $S_{\mathbf{c}}^{[k]}$ that are children

of r, i.e.,

$$S_{\mathbf{c}}^{[k]}(r) = \{s \in S_{\mathbf{c}}^{[k]} : s = ra \text{ for some } a \in \mathcal{A}\}. \tag{20}$$

Procedure RefineTypeClass iterates over nodes $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, and for each $s \in S_{\mathbf{c}}^{[k]}(r)$, it describes all potentially nonzero state transition counts $N_{s,t}$ with $t \in S_{\mathbf{c}}^{[k]}$. The actual encodings in the procedure will be done through an auxiliary procedure P, to be defined later in Figure 4. Given a node r and a symbol c, P(r, c) will describe $N_{s,t}$ for every $s \in S_{\mathbf{c}}^{[k]}(r)$, and every $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \text{head}(t)$. The correctness of the procedure is established in Lemma 12 below. The code length is analyzed later, when we prove Lemma 5. We make use of the following auxiliary lemma, whose proof is deferred to Appendix A.

Lemma 11: Let $h < k \leq d$. Then, $\left(\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right) = \mathcal{P}\left(\Delta T^{[k]}\right)$. Thus, by the conditions of Steps 2 and 6 of RefineTypeClass, we must have $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$ in Step 11.

Lemma 12: Assume that $\mathcal{K}_b(T, x^n)$, $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$, and $\sigma_{\mathbf{c}}(x^n)$ are known, and assume that P(r, c) correctly describes $N_{s,t}$ for every $s \in S_{\mathbf{c}}^{[k]}(r)$, and every $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \text{head}(t)$. Then, Procedure RefineTypeClass fully describes $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$.

Proof: The procedure iterates over all nodes $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$. Then, in each of the three cases distinguished by the conditions of Step 2 (nodes in $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, which, by (19) and the fact that $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, are in $S_{\mathbf{c}}^{[k-1]}$), Step 6 (nodes in $\mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$), and Step 11 (all remaining nodes $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$), its computations (possibly involving the use of Procedure P) allow the decoder to recover the counts for all state transitions that depart from every child of r that is a state of $T_{\mathbf{c}}^{[k]}$. Notice that the conditions of Steps 2 and 6 imply that $s \in S_{T^{[k]}}$ and $|s| = k$ in Step 12. Indeed, by Lemma 11, we must have $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$. Therefore, by Lemma 7, we have $|r| = k - 1$ and the child s of r is a state of $T^{[k]}$ at depth k. We next show that conditions required at various points of the computation are satisfied.

In Step 3 we ask for a symbol d such that $dr \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$. The condition $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ satisfied in Step 2 and the fact that $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, imply, by (19), that $r \in S_{\mathbf{c}}^{[k-1]}$. If $dr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ for all $d \in \mathcal{A}$, then r would be a forgetful state of $T_{\mathbf{c}}^{[k-1]}$, contradicting the definition of $T_{\mathbf{c}}^{[k-1]}$. Hence, the symbol d called for in Step 3 must exist.

In Step 5 we compute $N_{s,ds}$ as $n_s^{(d)}$, implicitly assuming that ds is a state of $T_{\mathbf{c}}^{[k]}$. Indeed, since $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ (as tested in Step 2), by Lemma 9, dr is an internal node of $T_{\mathbf{c}}^{[k]}$. However, by its definition in Step 3, dr is not an internal node of $T_{\mathbf{c}}^{[k-1]}$. Therefore, by Lemma 10(ii)–(iii), we must have $ds \in S_{\mathbf{c}}^{[k]}$.

The computations in Steps 5 and 14 require the knowledge of n_s. We claim that when picking an element r of $\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ in an iteration of Step 1, the description provided up to that point suffices for the decoder to reconstruct n_s for every $s \in S_{\mathbf{c}}^{[k]}(r)$. Consider a state $s \in S_{\mathbf{c}}^{[k]}(r)$, and let $v = \text{tail}(s)$. If $v \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, by Lemma 10(i), we have $v \in T_{\mathbf{c}}^{[k-1]}$, and with $a = \text{head}(s)$ we have $n_s = n_v^{(a)} + \delta$, which is known, given the final state in $T_{\mathbf{c}}$. Otherwise, if $v \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, all transitions into s come from the unique state s' of $T_{\mathbf{c}}^{[k]}$ such that $s' \preceq v$. Then, by Lemma 6, we have $n_s = N_{s',s} + \delta$, and, since s' is shorter than s, $N_{s',s}$ has already been computed in a previous iteration of the loop in Step 1, as the elements of $\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$

are taken in non-decreasing length order. Thus, the claim is proven, showing the validity of the decoding computations in Steps 5 and 14.

The fact that all states $s \in S_{\mathbf{c}}^{[k]}(r)$ in Step 12 belong to $T^{[k]}$ and have length k validates the use of $\mathcal{K}_b(T, x^n)$ to determine $n_s^{(a)}$ in Step 13, as well as the definition of s' in Steps 13 and 14, since the depth of $T_{\mathbf{c}}^{[k]}$ is k and, thus, $\overline{as}$ is sufficiently long to determine a state in $T_{\mathbf{c}}^{[k]}$ for every symbol a. ■

We next define Procedure P. The following lemma establishes a condition that will be assumed in the procedure.

Lemma 13: When P(r, c) is invoked in Step 4 or Step 10 of RefineTypeClass, the node r and the symbol c satisfy $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ and $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$.

Proof: The condition $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ is satisfied in both Step 4 and Step 10 since it is imposed in Step 1. Now, in Step 4, we have $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, since this condition is imposed in Step 2, and the claim follows from Lemma 9. In Step 10, the condition $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ is imposed in Step 9. ■

For $w \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, we denote by $\Delta T_{\mathbf{c}}^{[k]}(w)$ the subset of nodes in the canonical truncation increment, $\Delta T_{\mathbf{c}}^{[k]}$, that descend from w, i.e.,

$$\Delta T_{\mathbf{c}}^{[k]}(w) = \left\{t \in \Delta T_{\mathbf{c}}^{[k]} : w \preceq t\right\}, \quad h < k \leq d\,. \tag{21}$$

It is readily verified that the set of parent nodes of $\Delta T_{\mathbf{c}}^{[k]}(w)$, $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(w)\right)$, is, in fact, the subset of nodes in $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ that descend from w.

Procedure P is shown in Figure 4. In the procedure, b is an arbitrary but fixed symbol. Again, decoding steps are shown in brackets, to verify the losslessness of the code. The following lemma establishes the correctness of the procedure.

Lemma 14: Let $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ and $c \in \mathcal{A}$ such that $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ (see Lemma 13). Assuming that $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$ and $\sigma_{\mathbf{c}}(x^n)$ are known, P(r, c) correctly describes $N_{s,t}$ for every $s \in S_{\mathbf{c}}^{[k]}(r)$, and every $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \text{head}(t)$.

Proof: Since $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, all possibly non-zero state transition counts, $N_{s,t}$, such that $s \in S_{\mathbf{c}}^{[k]}(r)$ and $c = \text{head}(t)$, satisfy $\text{tail}(t) \not\preceq s$. Therefore, by (9), $s \preceq \text{tail}(t)$. We claim that all such state transition counts are correctly described.

We first analyze the case in which the condition in Step 1 is satisfied. In this case, since Step 1 checks that cr is a state of $T_{\mathbf{c}}^{[k-1]}$, the count n_{cr} required in Step 3 is known from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. Notice that, by the condition in Step 1, we have $r \in S_{\mathbf{c}}^{[k-1]}$ and $cr \in S_{\mathbf{c}}^{[k-1]}$, and also, by the assumptions, we have $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ and $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$. Therefore, for $d \in \mathcal{A}$, by Lemma 10(ii)–(iii), rd and crd belong to $S_{\mathbf{c}}^{[k]}$. Hence, all state transitions from $s \in S_{\mathbf{c}}^{[k]}(r)$ to $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \text{head}(t)$ must be of the form $s = rd$, $t = crd$, $d \in \mathcal{A}$, as implicitly assumed in Step 4. The reconstruction $N_{s,t}{=}n_{crd} + \delta$ in the same step follows from Lemma 6.

When the condition in Step 1 is not satisfied, the procedure starts a loop in Step 5, and, for each $s \in S_{\mathbf{c}}^{[k]}(r)$, it checks whether $cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$. If $cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, the procedure describes all state transition counts, $N_{s,t}$, with t of the form $t = csv$, $v \in \mathcal{A}^+$, distinguishing those destinations t that do not belong to

Procedure P(r, c)

Assumptions: $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$.

1. If r and cr are states of $T_{\mathbf{c}}^{[k-1]}$
2. Encode $\alpha-1$ counts n_{crd}, $d \in \mathcal{A}_b$.
3. [By (6), reconstruct $n_{crb} = n_{cr} - \sum_{d \neq b} n_{crd}$.]
   /* n_{cr} is known (see proof of Lemma 14). */
4. [Reconstruct $N_{s,t}$=$n_{crd} + \delta$ for all $d \in \mathcal{A}$, with $s = rd, t = crd$.]
   /* All state transitions from $s \in S_{\mathbf{c}}^{[k]}(r)$ to $t \in S_{\mathbf{c}}^{[k]}$ such that $c = head(t)$ are of the form $s = rd$, $t = crd$, $d \in \mathcal{A}$ (see proof of Lemma 14). */
5. else,
6. for each $s \in S_{\mathbf{c}}^{[k]}(r)$
7. If $cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$
8. For each $csu \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)$
9. Encode $\alpha-1$ counts n_{csud}, $d \in \mathcal{A}_b$.
10. [By (6), reconstruct $n_{csub} = n_{csu} - \sum_{d \neq b} n_{csud}$.]
    /* n_{csu} is known (see proof of Lemma 14). */
11. [Reconstruct $N_{s,t} = n_{csud} + \delta$ for all $d \in \mathcal{A}$, with $t = csud$.]
    /* $csud \in S_{\mathbf{c}}^{[k]}$ for all $d \in \mathcal{A}$ (see proof of Lemma 14). */
12. For each state $t = csv \notin \Delta T_{\mathbf{c}}^{[k]}(cs)$ of $T_{\mathbf{c}}^{[k]}$
13. [Reconstruct $N_{s,t} = n_{csv} + \delta$.]
    /* n_{csv} is known (see proof of Lemma 14). */
14. else
15. [Reconstruct $N_{s,t} = n_{cs}$+δ, for t=cs]
    /* $cs \in S_{\mathbf{c}}^{[k]}$ and n_{cs} is known (see proof of Lemma 14). */

Fig. 4. Encoding of state transition counts

$T_{\mathbf{c}}^{[k-1]}$ ($t \in \Delta T_{\mathbf{c}}^{[k]}(cs)$), from those that do belong to $T_{\mathbf{c}}^{[k-1]}$ ($t \notin \Delta T_{\mathbf{c}}^{[k]}(cs)$). Transition counts for those t that belong to $\Delta T_{\mathbf{c}}^{[k]}(cs)$ are described in the loop of Step 8, which iterates over $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)$, the set of all parent nodes of elements in $\Delta T_{\mathbf{c}}^{[k]}(cs)$. Now, if $csu \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)$, by (19), we have $csu \in S_{\mathbf{c}}^{[k-1]}$. Thus, the count n_{csu}, required in Step 10, is known from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. Moreover, by Lemma 10(ii) and (18), we have $csud \in S_{\mathbf{c}}^{[k]}$ for all $d \in \mathcal{A}$, as implicitly assumed in Step 11. The reconstruction $N_{s,t} = n_{csud} + \delta$ in Step 11 follows from Lemma 6. Transition counts for those t not in $\Delta T_{\mathbf{c}}^{[k]}(cs)$ are described in the loop of Step 12. By the definition of $\Delta T_{\mathbf{c}}^{[k]}(cs)$, if csv is a state of $T_{\mathbf{c}}^{[k]}$ such that $csv \notin \Delta T_{\mathbf{c}}^{[k]}(cs)$, then we must have $csv \in T_{\mathbf{c}}^{[k-1]}$. Hence, the value n_{csv} required in Step 13 is known from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. Again, Lemma 6 is used for the reconstruction in this step.

Finally, if $cs \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ in Step 7, then, by the assumption $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, we must have $cs \in S_{\mathbf{c}}^{[k]}$. Therefore, $N_{s,cs}$ is the only transition count departing from s that needs to be described, which, by Lemma 6, can be obtained as $N_{s,cs} = n_{cs}$+δ. Now, we must have either $cs \in T_{\mathbf{c}}^{[k-1]}$, or $s \in T_{\mathbf{c}}^{[k-1]}$, for

EnumCodeT (T, x^n)

1. Encode $\mathcal{K}(T, x^n)$ with EncodeTypeClass
2. Encode the index of x^n within $\mathcal{T}(T, x^n)$

Fig. 5. Enumerative code for tree models

otherwise, their respective parents cr and r, would belong to $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, which, by (19), implies that cr and r belong to $S_{\mathbf{c}}^{[k-1]}$ and Step 1 would not have branched to Step 5. Therefore, the count n_{cs} is known from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$, either directly or via $n_s^{(c)}$. ■

The overall scheme of an enumerative code for tree models is summarized in Figure 5. The enumeration of the type class for Step 2 is studied in [18]. In Subsection III-E we show that EnumCodeT can be implemented efficiently and that the code is universal for tree models. To prove the universality, in the next subsection, we will show that, using SCC, the encoding of each count that is explicitly described in Procedure P takes, on the average, $\frac{1}{2}\log n + O(1)$ bits. We finish the current subsection with a characterization of these counts, which will be useful for this purpose.

Lemma 15: All counts explicitly encoded in Procedure P are of the form n_t, where $t \in S_{\mathbf{c}}^{[k]}$ satisfies $s' \prec \mathrm{tail}(t)$ for some state s' *of* T.

Proof: All encodings take place in Steps 2 and 9. In Step 2, as argued in the proof of Lemma 14, $t = crd$ is a state of $T_{\mathbf{c}}^{[k]}$, which, since $T_{\mathbf{c}}^{[k]}$ is a tree of depth k, implies that $|r| < k-1$. Now, by the condition in Step 1, we have $r \in S_{\mathbf{c}}^{[k-1]}$ and, since $T^{[k-1]} \subseteq T_{\mathbf{c}}^{[k-1]}$, there exists a state s' of $T^{[k-1]}$ such that $s' \preceq r$. Moreover, since $|r| < k-1$ and $T^{[k-1]}$ is the truncation of T to depth $k-1$, s' must also be a state of T. Thus, $s' \prec \mathrm{tail}(t) = rd$ as claimed. Similarly, in Step 9, $t = csud$ is encoded, where, as argued in the proof of Lemma 14, $csud \in S_{\mathbf{c}}^{[k]}$. By the condition imposed in Step 7, we must have $|cs| < k$. Thus, by the condition in Step 6, s is a state of $T_{\mathbf{c}}^{[k]}$ and it satisfies $|s| < k$, which, again, implies that there exists a state s' in T that is a prefix of s. Thus, $s' \prec \mathrm{tail}(t) = sud$. ■

C. String count codes

We introduce string count codes (SCC), a class of non-uniform codes, which will be used in all explicit encodings in Procedure P and, also, as a tool to bound the expected size of the tree type classes in Subsection III-D. To this end, we describe counts n_t (as characterized in Lemma 15) by encoding the difference between n_t and an estimate of n_t based on values that will be known to the decoder. We start with some definitions and a lemma, and then describe the codes.

Let t be a string of length q such that $s' \prec \mathrm{tail}(t) = t_2^q$ for some state s' of T. By our assumption of T being non trivial, we assume $q > 2$. If $\bar{t}$ is a suffix of a string x^j, then, $\overline{\mathrm{tail}(t)}$ is a suffix of x^{j-1} and, thus, x^{j-1} must select the state s'. Further, let $\ell(t)$ be the largest integer such that each suffix $t_{\ell(t)-i+1}^q$

of t, $0 < i < \ell(t)$, has a (necessarily unique) state s_i as a proper prefix (at least $\ell(t) = 2$ satisfies this condition), i.e.,

$$\ell(t) = \max\{\, j : \ \forall i,\, 1 < i \le j,\ \exists s' \in S_T \ \text{ s.t. } \ s' \prec t_i^q \,\}\,, \tag{22}$$

where we notice that, since T is non trivial, we have $\ell(t) < q$. In the sequel, to lighten notation, we omit the dependence of $\ell(t)$ on t, and write simply ℓ. Then, the occurrence of $\bar{t}$ in x^n implies also the occurrence of $\overline{t_\ell^q}$, which selects state s_1, followed by the sequence of symbols $t_{\ell-1}, t_{\ell-2} \ldots t_1$, where each symbol $t_{\ell-i}$, $0 < i < \ell$, occurs in state s_i. Thus, with the short-hand notations

$$m_1 = n_{t_\ell^q}\,, \quad n_i = n_{s_i}\,, \quad n_i^\alpha = n_{s_i}^{(t_{\ell-i})}\,, \quad 0 < i < \ell\,, \tag{23}$$

we expect the estimate $m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}$ to be close to n_t for a typical sequence of the source. For example, in the tree T_1 of Figure 1, for $t = 100$, we have $q = 3$ and $\ell = 2$, since $t_3^3 = 0 \in T$. We have $s_1 = 0$, and the symbol $t_1 = 1$ occurs in state s_1 whenever $\bar{t}$ occurs in x^n.

The following lemma states that the estimate $m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}$ is available to the decoder when a count n_t is encoded in Procedure P.

Lemma 16: When a count n_t, for some $t \in \mathcal{A}^*$, is encoded in Procedure P, invoked from RefineTypeClass, the values m_1, n_i, and n_i^α, $0 < i < \ell$, are available to the decoder.

Proof: Procedure P is invoked from RefineTypeClass to encode some of the counts involved in the description of $\mathcal{K}(T_{\mathbf{c}}^{[k]}, x^n)$ given $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$, for some k, $h+2 \le k \le d$. By Lemma 15, these counts are of the form n_t, where t is a state of $T_{\mathbf{c}}^{[k]}$. Since, for $0 < i < \ell$, s_i is a state of T that is strictly shorter than t, and the length of t is at most the depth k of $T_{\mathbf{c}}^{[k]}$, we must have $s_i \in S_{T^{[k-1]}}$. Therefore, since $T^{[k-1]} \subseteq T_{\mathbf{c}}^{[k-1]}$, the counts n_i and n_i^α, $0 < i < \ell$, can be recovered from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. Similarly, since, by (22), $t_{\ell+1}^q$ belongs to T and it is strictly shorter than t, we also have $t_{\ell+1}^q \in T_{\mathbf{c}}^{[k-1]}$ and, thus, by (7), m_1 can be recovered from $\mathcal{K}(T_{\mathbf{c}}^{[k-1]}, x^n)$. ■

In view of Lemma 16, we next specify how we can efficiently describe n_t to a decoder that knows the values of m_1 and n_i^α, n_i, for all i, $0 < i < \ell$. Given m_1, we can estimate n_t by $m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}$, provided $n_i > 0$ for all i, $0 < i < \ell$ (otherwise we must have $n_t = 0$, which can be reconstructed by the decoder with no need for encoding). Assume the condition $n_i > 0$ holds for all i, $0 < i < \ell$, and define

$$z_t = n_t - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}\,, \quad Z_t = |z_t|\,, \quad \text{and} \quad \mathrm{sg}_t = \begin{cases} 1 & z_t > 0\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

As customary, denote by $\lfloor z \rfloor$ (resp. $\lceil z \rceil$) the largest (resp. smallest) integer satisfying $\lfloor z \rfloor \le z \le \lceil z \rceil$. Clearly,

$$n_t = \begin{cases} \lfloor Z_t \rfloor + \left\lceil m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} \right\rceil\,, & \mathrm{sg}_t = 1\,, \\ \left\lfloor m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} \right\rfloor - \lfloor Z_t \rfloor\,, & \text{otherwise}\,. \end{cases} \tag{24}$$

Hence, encoding $\lfloor Z_t \rfloor$ and sg_t suffices for the decoder to reconstruct n_t. Thus, we define a SCC for n_t, denoted $C_t(n_t)$, as follows:

- The sign sg_t is described plainly using one bit.
- The integer $\lfloor Z_t \rfloor$ is encoded using a Golomb code [19] of parameter $\lceil \sqrt{m_1} \rceil$. Specifically, we use a unary code for the integer division $\lfloor Z_t \rfloor^{\top} = \lfloor Z_t \rfloor \,/\, \lceil \sqrt{m_1} \rceil$, which takes $\lfloor Z_t \rfloor^{\top} + 1$ bits, and encode $\lfloor Z_t \rfloor^{\perp} = \lfloor Z_t \rfloor \mod \lceil \sqrt{m_1} \rceil$ uniformly with $\log\left(\lceil \sqrt{m_1} \rceil\right)$ bits. When $n_i = 0$ for some i, $0 < i < \ell$, or when $m_1 = 0$, we also have $n_t = 0$, and the decoder can reconstruct n_t with no need for encoding; we define $\lfloor Z_t \rfloor^{\top} = 0$ in this case, to simplify discussions on expectations.

The intuition behind the SCC $C_t(n_t)$ is that we can think of Z_t as the absolute difference between n_t and its estimate, $m_1 \prod_{i=1}^{\ell-1} \frac{n_i^{\alpha}}{n_i}$, which the decoder could guess from the known counters m_1 and n_i^{α}, n_i, for all i, $0 < i < \ell$. The probability of the estimate differing from the true value decays exponentially fast, which leads to a constant expectation of $\lfloor Z_t \rfloor^{\top}$, as stated in the following lemma.

Lemma 17: Let T be a tree model with all conditional probabilities different from zero. Then, the expectation $\mathrm{E}\left[\, \lfloor Z_t \rfloor^{\top} \,\right]$ is upper-bounded by a constant independent of n.

The proof of Lemma 17 is deferred to Appendix B.

Lemma 17 and the fact that $\log \lceil \sqrt{m_1} \rceil \le 1 + \frac{1}{2} \log n$ yield the following corollary.

Corollary 1: The expected code length of $C_t(n_t)$ is upper-bounded by $\frac{1}{2} \log n + O(1)$ bits.

D. The expected size of $\mathcal{T}(T, X^n)$.

In this subsection we study the asymptotic behavior of $\mathrm{E}\left[\, \log |\mathcal{T}(T, X^n)| \,\right]$ and prove Theorem 4. We first show how an economic description of $\mathcal{K}(T_F, x^n)$ can be obtained by means of SCC from one of $\mathcal{K}(T, x^n)$, where we recall that T_F denotes the FSM closure of T. This description, together with the bound (8) applied to T_F, and a coding argument, will lead to the bound claimed in Theorem 4.

Let

$$\kappa_T = -(|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}|) + |S_{T_F}|(\alpha - 1)\,. \tag{25}$$

By Theorem 3, the asymptotic number of type classes in T and T_F is $\Theta\left(n^{|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}|}\right)$ and $\Theta\left(n^{|S_{T_F}|(\alpha-1)}\right)$, respectively. Hence, by (25), there is, asymptotically, a factor of n^{κ_T} more type classes in T_F than in T, which suggests that roughly κ_T counts of $O(\log n)$ bits each would suffice to describe $\mathcal{K}(T_F, x^n)$ from $\mathcal{K}(T, x^n)$. Lemma 19 below confirms this intuition, and establishes the fact that the counts can be encoded economically. We make use of the following auxiliary Lemma 18, whose proof is deferred to Appendix C.

Lemma 18: For every tree T we have

$$\kappa_T = \sum_{v \in \mathcal{I}(T_F) \setminus \mathcal{I}(T_{\mathbf{c}})} (\alpha - 1)(\kappa_v - 1)\,, \tag{26}$$

where, for a node $v \in \mathcal{I}\,(T_F) \setminus \mathcal{I}\,(T_{\mathbf{c}})$, we let

$$\kappa_v = |\{a \in \mathcal{A} : at \notin \mathcal{I}\,(T_{\mathbf{c}})\}|\,. \tag{27}$$

Lemma 19: Given $\mathcal{K}(T, x^n)$, the collection $\mathcal{K}(T_F, x^n)$ can be described with κ_T counts n_t, $t \in \mathcal{A}^+$, plus a constant number of bits used to describe $\sigma_{\mathbf{r}}(x^n)$, the final state of x^n in T_F. Each of the κ_T counts, n_t, can be encoded using a SCC, $C_t(n_t)$, as defined in Section III-C, requiring $\frac{1}{2} \log n + O(1)$ bits.

Proof: Since $\sigma_{\mathbf{F}}(x^n)$ determines $\sigma_{\mathbf{c}}(x^n)$, by Lemma 3, $\mathcal{K}(T_{\mathbf{c}}, x^n)$ can be recovered from $\mathcal{K}(T, x^n)$ and $\sigma_{\mathbf{F}}(x^n)$. Hence, in order to recover $\mathcal{K}(T_F, x^n)$, it suffices to recover the counts $n_{vc}^{(a)}$ for every v in the set $D = \mathcal{I}(T_F) \setminus \mathcal{I}(T_{\mathbf{c}})$ and every $a, c \in \mathcal{A}$. By (7), any such count $n_{vc}^{(a)}$ can be reconstructed from one of the form n_t, $t = avc$, as $n_{vc}^{(a)} = n_t + \delta$, where we recall that δ is a (known) constant that accounts for border adjustments. We claim that n_t can be encoded using SCC. Indeed, since $v \in D$, then $v \notin \mathcal{I}(T)$ and, therefore, $\mathrm{tail}(t) = vc \notin T$, which implies that there exists some state s' of T that is a proper prefix of $\mathrm{tail}(t)$. In addition, since $\mathcal{K}(T, x^n)$ is known, all counts n_i, n_i^{α}, $0 < i < \ell$, are available to the decoder, where ℓ, n_i, and n_i^{α} are defined in (22) and (23). It remains to check that m_1, defined in (23), is also available. By (22), $t_{\ell+1}^{q}$ belongs to T and, thus, by (7), m_1 can be recovered from $\mathcal{K}(T, x^n)$.

Now, we claim that, if we describe these counts $n_{vc}^{(a)}$ taking the strings v from D in ascending order of $|v|$, n_{vc} can already be available at the time $n_{vc}^{(a)}$ is described. Let $v = bu \in D$, with $b \in \mathcal{A}$, and assume that, for all $w \in D$ such that $|w| < |v|$ and all $a, c \in \mathcal{A}$, $n_{wc}^{(a)}$ has already been recovered. Suppose first that $u \in \mathcal{I}(T_{\mathbf{c}})$. Then, $n_{uc}^{(b)}$ is known from $\mathcal{K}(T_{\mathbf{c}}, x^n)$ for every $c \in \mathcal{A}$ and, thus, by (7), n_{vc} can be reconstructed as $n_{vc} = n_{buc} = n_{uc}^{(b)} + \delta$. Suppose now that $u \notin \mathcal{I}(T_{\mathbf{c}})$. Since, by Theorem 1, every suffix of a state of T_F belongs to T_F [16] and $bu \in D \subseteq \mathcal{I}(T_F)$, we must have $u \in \mathcal{I}(T_F)$ and, therefore, $u \in D$. Hence, since u is shorter than v, by the assumed order of description, $n_{uc}^{(b)}$ is known for every $c \in \mathcal{A}$, and n_{vc} can be reconstructed as $n_{vc} = n_{buc} = n_{uc}^{(b)} + \delta$.

Next, we show how to recover the counts $n_{vc}^{(a)}$, for every $v \in D$ and every $a, c \in \mathcal{A}$, by explicitly describing a specific choice of κ_T of these counts. As shown, by taking the strings v from D in ascending order of length of v, n_{vc} is available at the time of describing $n_{vc}^{(a)}$. For every symbol a such that $av \in \mathcal{I}(T_{\mathbf{c}})$ and every $c \in \mathcal{A}$, since $avc \in T_{\mathbf{c}}$, n_{avc} is known from $\mathcal{K}(T_{\mathbf{c}}, x^n)$ and, therefore, we can reconstruct $n_{vc}^{(a)} = n_{avc} + \delta$. Thus, $n_{vc}^{(a)}$ requires no further description in this case. Now, since $v \in D$, there exists $s \in S_{T_{\mathbf{c}}}$ such that $s \preceq v$ and, since $T_{\mathbf{c}}$ is canonical, there exists a symbol $b \in \mathcal{A}$ such that $bs \notin \mathcal{I}(T_{\mathbf{c}})$, implying that, also, $bv \notin \mathcal{I}(T_{\mathbf{c}})$. Select one such symbol b. For each symbol c different from b, we describe $\kappa_v - 1$ counts, each of the form $n_{vc}^{(a)}$, with $a \neq b$ such that $av \notin \mathcal{I}(T_{\mathbf{c}})$ (by (27), there exist $\kappa_v - 1$ such symbols a). Since, by the claim above, n_{vc} is known, we can compute, by (4), $n_{vc}^{(b)} = n_{vc} - \sum_{a \neq b} n_{vc}^{(a)}$ and then, by (5), $n_{vb}^{(a)} = n_v^{(a)} - \sum_{c \neq b} n_{vc}^{(a)}$, for every $a \in \mathcal{A}$. Overall, we obtain $\mathcal{K}(T_F, x^n)$ from $\mathcal{K}(T_{\mathbf{c}}, x^n)$ and $\sigma_{\mathbf{F}}(x^n)$ by providing $(\alpha - 1)(\kappa_v - 1)$ counts for each $v \in D$, which, by Lemma 18, sum up to κ_T counts. ∎

Next, we apply the results of Lemma 19 in a coding argument to complete the proof of Theorem 4.

Proof of Theorem 4: A sequence in $\mathcal{T}(T, x^n)$ can be encoded by describing the subset $\mathcal{T}(T_F, x^n)$ to which the sequence belongs, and then, uniformly, its index within that subset. The expected code length is upper-bounded using Lemma 19 and is lower-bounded by the conditional entropy of the sequence given its type with respect to T, yielding

$$\mathrm{E}\left[\log|\mathcal{T}(T, X^n)|\right] \leq \mathrm{E}\left[\log|\mathcal{T}(T_F, X^n)|\right] + \frac{1}{2}\kappa_T \log n + O(1)\,, \tag{28}$$

where we use the fact that the sequences in $\mathcal{T}(T, X^n)$ are equiprobable. Applying Lemma 1 to T_F, we

obtain

$$\frac{1}{n}\mathrm{E}\left[\log|\mathcal{T}(T_F,X^n)|\right] \le \mathcal{H} - \frac{|S_{T_F}|(\alpha-1)}{2n}\log n + O\left(\frac{1}{n}\right),$$

which, together with (28) and Lemma 18, yields

$$\frac{1}{n}\mathrm{E}\left[\log|\mathcal{T}(T,X^n)|\right] \le \mathcal{H} - \frac{|E_{T_\mathbf{c}}|-|V_{T_\mathbf{c}}|}{2n}\log n + O\left(\frac{1}{n}\right). \qquad \blacksquare$$

E. Universality and efficiency of EnumCodeT

In this subsection we show that the proposed enumerative code is universal for tree models and it can be efficiently implemented, thus proving Theorem 5. We also present an alternative enumerative code, which is also universal for a given tree T, although the enumeration of sequences is with respect to type classes of T_F, the FSM closure of T. We start by analyzing the number of counts that are explicitly encoded in Steps 3–4 of EncodeTypeClass. We make use of the following auxiliary lemma, whose proof is deferred to Appendix D.

Lemma 20: The number of counts described in the iteration corresponding to k, $h+2 \le k \le d$, in Steps 3-4 of EncodeTypeClass (Figure 2) is

$$C_k = \left(|E_{T_\mathbf{c}^{[k]}}| - |V_{T_\mathbf{c}^{[k]}}|\right) - \left(|E_{T_\mathbf{c}^{[k-1]}}| - |V_{T_\mathbf{c}^{[k-1]}}|\right) - (\alpha-1)\left(|S_{T^{[k]}}| - |S_{T^{[k-1]}}|\right).$$

The total number of counts that are actually encoded is stated in the following lemma.

Lemma 21: The number of counts encoded by EncodeTypeClass as the algorithm iterates through the loop in Steps 3-4 is $(|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|) - |S_T|(\alpha-1)$.

Proof: By Lemma 20, the total number of counts described by EncodeTypeClass as the algorithm iterates through the loop in Steps 3-4 is

$$\sum_{k=h+2}^{d} C_k = \sum_{k=h+2}^{d}\left[\left(|E_{T_\mathbf{c}^{[k]}}| - |V_{T_\mathbf{c}^{[k]}}|\right) - \left(|E_{T_\mathbf{c}^{[k-1]}}| - |V_{T_\mathbf{c}^{[k-1]}}|\right) - (\alpha-1)\left(|S_{T^{[k]}}| - |S_{T^{[k-1]}}|\right)\right]. \tag{29}$$

Recalling that $T^{[d]} = T$ and $T_\mathbf{c}^{[d]} = T_\mathbf{c}$, the telescopic sum in (29) reduces to

$$(|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|) - \left(|E_{T_\mathbf{c}^{[h+1]}}| - |V_{T_\mathbf{c}^{[h+1]}}|\right) - (\alpha-1)\left(|S_T| - |S_{T^{[h+1]}}|\right). \tag{30}$$

Now, since $T^{[h+1]}$ is FSM, $V_{T_\mathbf{c}^{[h+1]}} = V_{T^{[h+1]}} = S_{T^{[h+1]}}$, $E_{T_\mathbf{c}^{[h+1]}} = E_{T^{[h+1]}}$, and $|E_{T^{[h+1]}}| = \alpha|S_{T^{[h+1]}}|$, so that $|E_{T_\mathbf{c}^{[h+1]}}| - |V_{T_\mathbf{c}^{[h+1]}}| = (\alpha-1)|S_{T^{[h+1]}}|$ and (30) becomes

$$(|E_{T_\mathbf{c}}| - |V_{T_\mathbf{c}}|) - (\alpha-1)|S_T|. \qquad \blacksquare$$

The number of counts in Lemma 21 is precisely the one sought in Lemma 5 and, by Lemmas 15 and 16, SCC can be applied for the encoding of counts in Procedure P, yielding the claim on code length of Lemma 5. To prove the claim on algorithmic complexity we will make use of the following lemma.

Lemma 22: For a tree T we have $|T_{\mathbf{c}}| < |E_{T_{\mathbf{c}}}| = |E_T| = O(|S_T|)$.

Proof: Since $T_{\mathbf{c}}$ is a full tree, we have $|T_{\mathbf{c}}| = (\alpha|S_{T_{\mathbf{c}}}|-1)/(\alpha-1) < \alpha|S_{T_{\mathbf{c}}}|$. Therefore, since there are at least α edges in $E_{T_{\mathbf{c}}}$ departing from each state of $|S_{T_{\mathbf{c}}}|$, we have $|T_{\mathbf{c}}| < |E_{T_{\mathbf{c}}}|$. Now, if we obtain a tree T' by extending a forgetful state s of a tree $\tilde{T}$, since the edges of $E_{\tilde{T}}$ departing from s are in one-to-one correspondence with the edges of $E_{T'}$ departing from the children of s in T', we have $|E_{T'}| = |E_{\tilde{T}}|$. Thus, we must have $|E_{T_{\mathbf{c}}}| = |E_T|$. Finally, notice that the number of pairs (s,t) in (9) satisfying $s \preceq \mathrm{tail}(t)$ is at most $|S_T|$ (at most one for each t), and at most $\alpha|S_T|$ pairs satisfy $\mathrm{tail}(t) \prec s$ (at most α for each s). Therefore, we have $|E_T| = O(|S_T|)$. ■

Proof of Lemma 5: In Step 1, EncodeTypeClass describes $\mathcal{K}_b(T, x^n)$ using $|S_T|(\alpha - 1)$ counts of $\log n$ bits each, and the final state of x^n in $T_{\mathbf{c}}$ using a constant number of bits. In the loop of Steps 3–4, EncodeTypeClass completes the encoding of $\mathcal{K}(T, x^n)$ by describing an additional set of counts using SCC. By Lemma 21, SCC are used $(|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}|) - (\alpha-1)|S_T|$ times taking, by Corollary 1, an average of at most $\frac{1}{2}\log n + O(1)$ bits each.

Turning to the computational complexity claim, we first observe that, by Lemma 4, the decoder can efficiently reconstruct $\mathcal{K}(T, x^n)$ given $\mathcal{K}_b(T, x^n)$. Thus, it remains to check that the loop in Steps 3–4 of EncodeTypeClass, and the corresponding decoding steps, can be performed with a number of operation that is polynomial in n and $|T|$. By Lemma 22, we can equivalently check that the number of required operations is polynomial in n and $|T_{\mathbf{c}}|$.

By Lemma 8 and Lemma 10(i), constructing $T_{\mathbf{c}}^{[k]}$ given $T_{\mathbf{c}}^{[k-1]}$ ammounts to checking, for each state s of $T_{\mathbf{c}}^{[k-1]}$, wheteher s is forgetfull in $T_{\mathbf{c}}^{[k]}$ and, if so, adding a full complement of children to s. Therefore, since $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}$, the construction of the sequence of canonical trees, $T_{\mathbf{c}}^{[h+1]} \dots T_{\mathbf{c}}^{[d]}$, requires checking wheteher a state is forgetfull a number of times that is upper-bounded by $d|S_{T_{\mathbf{c}}}|$, where $d < |T_{\mathbf{c}}|$. Thus, it remains to verify that for each iteration k of the loop in Step 3, the invokation to RefineTypeClass in Step 4 requires a number of operation that is polynomial in n and $|T_{\mathbf{c}}|$.

RefineTypeClass consists of a loop that iterates over nodes $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ in non-decreasing order of length, where, by definition, $\left|\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)\right| < |T_{\mathbf{c}}^{[k]}|$. Except for invocations to Procedure P, it is straightforward to verify that all other operations executed by RefineTypeClass in an iteration of the loop in Step 1 take polynomial time in $|T_{\mathbf{c}}^{[k]}|$, where we recall that $T_{\mathbf{c}}^{[k]} \subseteq T_{\mathbf{c}}$. Procedure P involves, possibly, a partial traversal of the tree $T_{\mathbf{c}}^{[k]}$ in the loops of Step 8 and Step 12, and encoding $C_t(n_t)$, for all t in certain subset of $S_{\mathbf{c}}^{[k]}$. Encoding $C_t(n_t)$ requires calculating the estimate z_t, which takes $O(d)$ multiplications, plus operations that do not depend on $|T|$, associated to the Golomb encoding of $\lfloor Z_t \rfloor$, which can be executed in time that is polynomial in n. ■

Proof of Theorem 5: Applying Lemma 5 and Theorem 4 to bound the expected code length of Steps 1 and 2 of EnumCodeT, respectively, we arrive at (13). Moreover, by Lemma 5 and Theorem 2, Steps 1 and 2, respectively, require a computation time polynomial in $|T|$ and n. ■

The foregoing results yield the following corollary.

Corollary 2: Let T be a tree model with all conditional probabilities different from zero. Then, for a random sequence X^n emitted by the corresponding source, the type class of X^n relative to the FSM closure, T_F, of T, can be described using, on average, $\frac{1}{2}(\alpha-1)\left(|S_T|+|S_{T_F}|\right)\log n + O(1)$ bits.

Proof: By Lemma 19, given a description of $\mathcal{K}(T,x^n)$, we can obtain one of $\mathcal{K}(T_F,x^n)$ with a cost of $\frac{1}{2}\kappa_T \log n + O(1)$ additional bits on average, where we recall the definition of κ_T in (25). Accounting for the cost of the description of $\mathcal{K}(T,x^n)$ from (12) yields the claimed result. ■

The result of Corollary 2 suggests the following alternative enumerative coding strategy: To encode x^n, encode $\mathcal{K}(T_F,x^n)$ as described in the corollary, and then describe the index of x^n in an enumeration of $\mathcal{T}(T_F,x^n)$. Using the bound of Lemma 1, applied to T_F, for the expected code length of this index, we obtain an upper bound on the expected total normalized code length of $\mathcal{H} + \frac{1}{2}(\alpha-1)|S_T|\frac{\log n}{n} + O(1/n)$. Notice that although the enumeration is done on $\mathcal{T}(T_F,x^n)$, the expected redundancy is still optimal with respect to the smaller model T. Since enumerating $\mathcal{T}(T_F,x^n)$ is simpler than enumerating $\mathcal{T}(T,x^n)$ when T is not FSM, this alternative yields a simpler implementation of the second part of an enumerative code, at the cost of a more complex description of the type class.

IV. Twice-Universal Coding

In this section we switch to a twice-universal setting in which the tree T is unknown. Our first approach follows a conceptually simple, standard plug-in strategy in which we estimate a tree $\hat{T}$ and then use EnumCodeT with $\hat{T}$ as if it were the true tree underlying the model. Later, we will demonstrate an alternative approach in which EnumCodeT can be simplified for the twice-universal setting, at the cost of an increment of redundancy of order $\log\log n$, which is negligible with respect to the main redundancy term.

A. Plug-in approach

A tree T and a sequence x^n determine an (empirical) probability distribution on sequences of length n, $\hat{P}_T$, defined by T and a model parameter $\hat{p}_T(a|s) = n_s^{(a)}(x^n)/n_s(x^n)$ (as before, we omit the dependence of the distribution $\hat{P}_T$ on x^n when clear from the context); $\hat{P}_T(x^n)$ is the maximum-likelihood probability of x^n under T. We consider a class of penalized maximum-likelihood tree model estimators. Specifically, given a sequence x^n, we assign to a tree T a cost $K(T,x^n) = -\log \hat{P}_T(x^n) + C(T)\log n$, where the *penalization function* $C(T)$ is increasing with $|S_T|$. We have

$$K(T,x^n) = -\sum_{s\in S_T, a\in\mathcal{A}} n_s^{(a)} \log \frac{n_s^{(a)}}{n_s} + C(T)\log n\,.$$

The tree estimate $\hat{T}(x^n)$ for x^n is defined as the tree that minimizes the cost function $K(T,x^n)$ over all possible trees, that is,

$$\hat{T}(x^n) = \arg\min_T \{K(T,x^n)\}\,. \tag{31}$$

Efficient algorithms are known for finding the minimizing tree $\hat{T}(x^n)$; see, e.g., [16], [21].

Twice-EnumCodeT(x^n)

1. Compute the estimate $\hat{T}(x^n)$ of T.
2. Describe $\hat{T}$ to the decoder.
3. Encode x^n using EnumCodeT with respect to the tree $\hat{T}$.

Fig. 6. Twice universal enumerative code for tree models

We define the code *Twice-EnumCodeT* algorithmically in Figure 6.[8]

Using a natural code [22] for describing the full tree $\hat{T}(x^n)$, Step 2 of Twice-EnumCodeT requires one bit per node. To estimate the cost of Step 3, we must analyze the code length of EnumCodeT when applied to $\hat{T}(x^n)$ rather than T. The analysis will rely on upper bounds on the probabilities of over-estimation and under-estimation of T, which are stated in the two lemmas below. Similar bounds are well known for several estimators. Proofs for the lemmas can be readily adapted from [13], and are omitted here.

Lemma 23: Let T be a tree model and consider a penalization function of the form $C(T) = \beta|S_T|$ with $\beta > \frac{\alpha(\alpha+1)+1}{\alpha-1}$. Let $O^n \subseteq \mathcal{A}^n$ be the set of strings for which a state of T is internal to the estimated tree $\hat{T}$. Then, for a random sequence X^n emitted by the source modeled by T, we have $\mathrm{P}_T\{X^n \in O^n\} \leq |S_T| n^{-\gamma}$ with $\gamma = \beta(\alpha-1) - \alpha(1+\alpha) - 1 > 0$.

Lemma 24: Let T be a minimal tree model and consider a penalization function of the form $C(T) = \beta|S_T|$. Let $U^n \subseteq \mathcal{A}^n$ be the set of sequences whose estimated tree $\hat{T}$ has a state that is internal to T. Then, for a random sequence X^n emitted by the source modeled by T, we have $\mathrm{P}_T\{X^n \in U^n\} \leq \rho 2^{-n\gamma}$ for positive constants ρ, γ.

It follows from Lemma 23 and Lemma 24 that we can choose β to make the contribution of sequences with estimated tree $\hat{T} \neq T$ to the expected code length negligible, provided that the code length is upper-bounded by a polynomial in n for every sequence of length n. We show such a bound in the proof of the following theorem.

Theorem 6: Let T be a tree model with all conditional probabilities different from zero and let $\mathcal{H}$ be the entropy rate of the corresponding source. Then, taking a penalization function $C(T) = \beta|S_T|$ with β sufficiently large, the normalized expected code length of Twice-EnumCodeT satisfies

$$\mathrm{E}\left[\frac{L(X^n)}{n}\right] \leq \mathcal{H} + \frac{|S_T|(\alpha-1)\log n}{2n} + O\left(\frac{1}{n}\right).$$

Proof: It suffices to show that $L(x^n)$ is upper-bounded by a polynomial in n for every sequence x^n.

[8]The enumeration of type classes presented in [18] requires fixing an initial state s_0 of maximal depth in the tree. Since $\hat{T}$ is not known in advance, the leading string $x^0_{-\infty}$ used to determine the empirical probabilities upon which the estimate $\hat{T}$ is based in (31) may turn out to select a state that is not of maximal depth in $\hat{T}$. Therefore, for the purpose of Step 3, we will derive counts based on an initial state $\hat{s}_0$ of maximal depth in $\hat{T}$, selected with some fixed deterministic policy (say, the smallest lexicographically among maximal depth states). These counts may be different from those used for estimating $\hat{T}$.

In Twice-EnumCodeT we describe $\mathcal{K}(\hat{T}, x^n)$ by encoding the final state of x^n with respect to $\hat{T}_{\mathbf{c}}$, encoding $\mathcal{K}_b(\hat{T}, x^n)$ with $|S_{\hat{T}}|(\alpha-1)$ counts of $\log n$ bits each, and finally giving, by Lemma 21, an additional set of $|E_{\hat{T}_{\mathbf{c}}}|-|V_{\hat{T}_{\mathbf{c}}}|-|S_{\hat{T}}|(\alpha-1)$ counts described with SCC, which take $O(\sqrt{n})$ bits each. Thus, the complete description of $\mathcal{K}(\hat{T}, x^n)$ takes $O\left(\left(|E_{\hat{T}_{\mathbf{c}}}| - |V_{\hat{T}_{\mathbf{c}}}|\right)\sqrt{n}\right)$ bits, or, by Lemma 22, $O(|S_{\hat{T}}|\sqrt{n})$ bits. Since the index of x^n within its class takes no more than n bits, we upper-bound $L(X^n)$ by $O\left(|S_{\hat{T}}|\sqrt{n}+n\right)$.

To obtain the desired polynomial bound on the total code length, it remains to bound $|S_{\hat{T}}|$. Since $\hat{T}$ minimizes $K(T, x^n)$, comparing it with the single node tree (i.e., a zero-order model) we get,

$$-\log \hat{P}_{\hat{T}}(x^n) + \beta|S_{\hat{T}}|\log n \le n\hat{\mathcal{H}}(x^n) + \beta \log n\,,$$

where $\hat{\mathcal{H}}(x^n)$ is the zero-order empirical entropy of x^n. Since $-\log \hat{P}_{\hat{T}}(x^n) \ge 0$ and $\hat{\mathcal{H}}(x^n) \le \log\alpha$, we conclude that $|S_{\hat{T}}| = O(n/\log n)$. Thus, we have $L(X^n) = O(n^{3/2})$. ■

B. Simplified scheme

We present an alternative code EnumCodeT′, which is a simplification of EnumCodeT, applicable when the target tree is an estimate $\hat{T}$, as in Step 3 of Twice-EnumCodeT. Recall that a fundamental tool in EnumCodeT is the use of SCC for encoding certain counts, which, by Corollary 1, require $\frac{1}{2}\log n + O(1)$ bits on average each. The following lemma leads to an analogue of Corollary 1 in the twice-universal setting, for which we recall the definitions of ℓ, m_1, n_i, and n_i^α from (22) and (23).

Lemma 25: Let $\hat{T}$ be a tree estimate for x^n according to (31) with a penalization function $C(T) = \beta|S_T|$, and let t be a string such that $s' \prec \mathrm{tail}(t)$ for some $s' \in S_{\hat{T}}$ and $n_i > 0$ for all i, $0 < i < \ell$. Then,

$$|z_t| = \left| n_t - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} \right| \le (\ell-1)\left(\sqrt{2\beta\alpha|t|n_s \ln n}+1\right)\,. \tag{32}$$

The assumptions on t of Lemma 25 are required conditions for the application of a SCC, $C_t(n_t)$, to encode n_t, as defined in Section III-C, to a decoder that knows the values of m_1, n_i^α, and n_i, for all i, $0 < i < \ell$. Clearly, z_t suffices for such a decoder to recover n_t. Thus, by Lemma 25, we can replace $C_t(n_t)$ with a uniform code $U_t(n_t)$ for z_t in the range given by (32), when applying EnumCodeT to encode x^n with respect to $\hat{T}$. We prove Lemma 25 in Appendix E. The following corollary is an immediate consequence of the lemma.

Corollary 3: Let $\hat{T}$ be a tree estimate for x^n according to (31) with a penalization function $C(T) = \beta|S_T|$, and let t be a string satisfying the assumptions of Lemma 25. Then, the code length of $U_t(n_t)$ is $\frac{3}{2}\log|t| + \frac{1}{2}\log n + O(\log\log n)$ bits.

Proof: Since $\ell < |t|$ and $n_s \le n$ in (32), we have

$$|z_t| \le |t|^{3/2}\left(\sqrt{2\beta\alpha n\ln n} + |t|^{-1/2}\right) \le |t|^{3/2}\left(\sqrt{2\beta\alpha n\ln n}+1\right)\,. \qquad ■$$

Corollary 3 is an analogue of Corollary 1 in the twice-universal setting. Notice, however, that the upper bound here is *pointwise*, and not just in expectation.

We now define the code *EncodeTypeClass′* exactly as EncodeTypeClass, but replacing the use of SCC, $C_t(n_t)$, by an encoding of z_t with $U_t(n_t)$. We also define *EnumCodeT′* as the code obtained by substituting EncodeTypeClass′ for EncodeTypeClass in EnumCodeT and, analogously, we define *Twice-EnumCodeT′* as the code obtained by substituting EnumCodeT′ for EnumCodeT in Twice-EnumCodeT.[9] These variants take advantage of the idea suggested in Section I, namely, that for some type classes of $\hat{T}$, no sequence in the class will estimate $\hat{T}$. Therefore, these "atypical" classes can be excluded from the coding space, which is reflected in the limited range for the uniform encoding of z_t with $U_t(n_t)$. The result is a twice-universal enumerative code for tree models, as stated in the following theorem.

Theorem 7: Let T be a tree model with all conditional probabilities nonzero and let $\mathcal{H}$ be the entropy rate of the corresponding source. Estimating $\hat{T}(x^n)$ according to (31) with a penalization function $C(T) = \beta|S_T|$ for sufficiently large β, the normalized expected code length of Twice-EnumCodeT′, for a random sequence X^n emitted by this source, satisfies

$$\mathrm{E}\left[\frac{L(X^n)}{n}\right] \leq \mathcal{H} + \frac{|S_T|(\alpha-1)\log n}{2n} + O\left(\frac{\log\log n}{n}\right). \tag{33}$$

Proof: In EncodeTypeClass, SCC are applied to encode occurrence counts of states of the canonical extension of a truncation of T. Hence, when we use $U_t(n_t)$ in EncodeTypeClass′, $|t|$ is bounded by the depth of $\hat{T}(x^n)$ (since, by Lemma 2, for any tree T, $T_{\mathbf{c}}$ and T have the same depth). For sequences that estimate $\hat{T}(x^n) = T$, $|t|$ is bounded by the depth of T (a constant), and the encoding of z_t with $U_t(n_t)$, by Corollary 3, takes $\frac{1}{2}\log n + O(\log\log n)$ bits. Thus, for sequences in the set $\Phi = \{x^n : \hat{T}(x^n) = T\}$, the code length of EncodeTypeClass′ satisfies

$$\begin{aligned}\Big|\text{EncodeTypeClass}'(\hat{T}, x^n)\Big| \leq \\ \leq \quad (\alpha-1)|S_T|\log n + \Big((|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}|) - (\alpha-1)|S_T|\Big)\frac{1}{2}\log n + O(\log\log n)\,,\end{aligned} \tag{34}$$

where the first term in (34) corresponds to the description of $\mathcal{K}_b(T, x^n)$ in Step 1 in Figure 2, and the second and third terms to the encodings $U_t(n_t)$, which, by Lemma 21, are used $(|E_{T_{\mathbf{c}}}| - |V_{T_{\mathbf{c}}}|) - (\alpha-1)|S_T|$ times taking, by Corollary 3, $\frac{1}{2}\log n + O(\log\log n)$ bits each. Thus, applying Theorem 4 and using (34), we conclude that the right-hand side of (33) is an upper bound on the contribution of sequences in Φ to the normalized expected code length of Twice-EnumCodeT′. To complete the proof, by Lemma 23 and Lemma 24, it suffices to show that $L(x^n)$ is upper-bounded by a polynomial in n for every sequence x^n.

[9] As with Twice-EnumCodeT, Twice-EnumCodeT′ requires a fixed deterministic policy for the selection of the initial state, since, in principle, the leading string $x_{-\infty}^0$ may turn out to select a state that is not of maximal depth in $\hat{T}$. In this case, a count $n_w^{(a)}$ may change to $\tilde{n}_w^{(a)}$ after fixing the new initial conditions, and the range of $\tilde{n}_w^{(a)}$ is not necessarily given by (32). It is not difficult to adapt Lemma 25 to yield a suitable range for $\tilde{n}_w^{(a)}$. Alternatively, we can describe $\mathcal{T}(\hat{T}, x^n)$ by giving the original counts $\mathcal{K}(\hat{T}, x^n)$ with respect to $x_{-\infty}^0$, together with additional information that allows the reconstruction of the counts that result after fixing the new initial conditions. For example, we could use $\log|\mathcal{I}\left(\hat{T}\right)| + \log\alpha$ bits to describe the longest internal node of $\hat{T}$, u, such that $\overline{u} \prec x^n$, and the symbol $x_{|u|+1}$. This additional information does not affect the asymptotic normalized expected code length of the algorithm.

This follows from the fact that, by Corollary 3, each encoding $U_t(n_t)$ takes $\frac{1}{2}\log n + \frac{3}{2}\log|t| + O(\log\log n)$ bits, recalling that $|t|$ is upper-bounded by the depth of $\hat{T}$, and using the same arguments as in Theorem 6. ■

Notice that, since Twice-EnumCodeT$'$ is universal with respect to any fixed tree model T such that all conditional probabilities are different from zero, Twice-EnumCodeT$'$ could be used in place of EnumCodeT even if T is known, taking advantage of the simplification of $U_t(n_t)$ with respect to $C_t(n_t)$. This simplification, however, would come at the cost of unnecessarily estimating $\hat{T}$ and a penalty on the redundancy of order $O(\log\log n)$.

APPENDIX A
PROOFS OF LEMMAS 8, 10 AND 11

Proof of Lemma 8: Suppose that $T_{\mathbf{c}}^{[k-1]} \not\subseteq T_{\mathbf{c}}^{[k]}$. We recall, from Subsection II-C, that the minimal canonical extension of a tree is obtained by sequentially extending forgetful states until no such state is left. Let $v_1, v_2, \ldots, v_m$ be the sequence of forgetful states extended in the process of constructing $T_{\mathbf{c}}^{[k-1]}$ from $T^{[k-1]}$. There must be some node in this sequence that is not in $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ (otherwise we would have $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$). Let v_j be the first such node in the sequence. We must have $v_j \in S_{\mathbf{c}}^{[k]}$, for otherwise v_j would not belong to $T_{\mathbf{c}}^{[k]}$ and, therefore, since $T^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$, v_j would have been created by extending some v_i, with $i < j$ such that $v_i \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, contradicting the definition of j. Let T_j' be the tree in the sequence of extensions leading to $T_{\mathbf{c}}^{[k-1]}$ at the time v_j was chosen for extension. Since v_j is forgetful in T_j' and, by the definition of v_j, $T_j' \subseteq T_{\mathbf{c}}^{[k]}$, v_j must also be a forgetful state in $T_{\mathbf{c}}^{[k]}$, in contradiction to the fact that this tree is canonical. Thus, we must have $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$. ■

We next prove two auxiliary lemmas and then proceed to prove Lemma 10 and Lemma 11.

Lemma 26: For $h < k \leq d$, the set of parents of nodes in the canonization increment, $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, satisfies $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq S_{\mathbf{c}}^{[k-1]}$.

Proof: Suppose the claim is not true, i.e., $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \not\subseteq S_{\mathbf{c}}^{[k-1]}$ for some k, $h < k \leq d$. By (17), every element of $\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$ is of the form su, with $s \in S_{\mathbf{c}}^{[k-1]}$ and $u \in \mathcal{A}^+$, where, by our assumption of T being non-trivial, we must have $|s| \geq 1$. Therefore, $\lambda \notin \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ and, thus, we can pick $z \in \mathcal{A}^*$ and $c \in \mathcal{A}$ such that $zc \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ is of maximal length among those elements of $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ that are not states of $T_{\mathbf{c}}^{[k-1]}$. By Lemma 9, we know that $azc \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ for every $a \in \mathcal{A}$. We claim that $az \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ for every $a \in \mathcal{A}$. If $azc \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, since azc is longer than zc, azc is a state of $T_{\mathbf{c}}^{[k-1]}$. Therefore, $az \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$. If, on the other hand, $azc \notin \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, then, for $d \in \mathcal{A}$, we have $azcd \notin \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$ and also, since $azc \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, we have $azcd \in T_{\mathbf{c}}^{[k]}$. Hence, by the definition of the canonization increment, $\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$, in (17), either $azcd \in T_{\mathbf{c}}^{[k-1]}$ or $azcd \in T^{[k]}$. In either case $azc \in T_{\mathbf{c}}^{[k-1]}$, i.e., az is an internal node of $T_{\mathbf{c}}^{[k-1]}$. We conclude that $az \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ for every $a \in \mathcal{A}$, as claimed. As a consequence, $z \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$, for otherwise the state $s \in S_{\mathbf{c}}^{[k-1]}$ such that $s \preceq z$ would be forgetful in $T_{\mathbf{c}}^{[k-1]}$. This implies that $zc \in T_{\mathbf{c}}^{[k-1]}$. Furthermore, since $zc \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, $zcd \in \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$ for some $d \in \mathcal{A}$. Therefore, $zcd \notin T_{\mathbf{c}}^{[k-1]}$, implying that zc is a state of $T_{\mathbf{c}}^{[k-1]}$, a contradiction. ■

Lemma 27: For $h < k \leq d$, $\{\mathcal{P}\left(\Delta T^{[k]}\right), \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)\}$ is a partition of $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$.

Proof: Since, by (15) and (17), the truncation increment, $\Delta T^{[k]}$, and the canonization increment, $\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$, are disjoint, then $\mathcal{P}\left(\Delta T^{[k]}\right)$ and $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ must also be disjoint, for otherwise, there would exist some $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$ and symbols $a, b \in \mathcal{A}$, such that $ra \in \Delta T^{[k]}$ and $rb \in \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$, implying that $ra \in T^{[k]}$ but $rb \notin T^{[k]}$, contradicting the fact that $T^{[k]}$ is a full tree. Therefore, we need to show that $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) = \mathcal{P}\left(\Delta T^{[k]}\right) \cup \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, for which it suffices to show that $\Delta T_{\mathbf{c}}^{[k]} = \Delta T^{[k]} \cup \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$, or, since $\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]} \subseteq \Delta T_{\mathbf{c}}^{[k]}$, that $\Delta T^{[k]} = \Delta T_{\mathbf{c}}^{[k]} \setminus \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$. Now, if $u \in \Delta T_{\mathbf{c}}^{[k]} \setminus \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$, then, by the definitions (17) and (16), $u \in T^{[k]} \setminus T_{\mathbf{c}}^{[k-1]}$, which, since $T^{[k-1]} \subseteq T_{\mathbf{c}}^{[k-1]}$, implies that $u \in \Delta T^{[k]}$. Conversely, if $u \in \Delta T^{[k]}$, by Lemma 7, we have $|u| = k$ and, therefore, since $T_{\mathbf{c}}^{[k-1]}$ has depth $k-1$, $u \notin T_{\mathbf{c}}^{[k-1]}$. Thus, we have $u \in \Delta T_{\mathbf{c}}^{[k]}$ and, furthermore, since $u \in T^{[k]}$, we must have $u \in \Delta T_{\mathbf{c}}^{[k]} \setminus \Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}$.

■

Proof of Lemma 10: (i) We claim that $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) \subseteq S_{\mathbf{c}}^{[k-1]}$. By Lemma 26, we have $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq S_{\mathbf{c}}^{[k-1]}$. On the other hand, if $r \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, by Lemma 27, we have $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$, which, by Lemma 7, implies that $r \in S_{T^{[k-1]}}$ and $|r| = k-1$. Thus, since $T_{\mathbf{c}}^{[k-1]}$ has depth $k-1$, we must also have $r \in S_{\mathbf{c}}^{[k-1]}$. We conclude that $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) \subseteq S_{\mathbf{c}}^{[k-1]}$. Moreover, since, by definition, $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ is a subset of $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, we have

$$\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) \subseteq S_{\mathbf{c}}^{[k-1]} \setminus S_{\mathbf{c}}^{[k]}\,. \tag{35}$$

If $u \in \Delta T_{\mathbf{c}}^{[k]}$, then we must have $u \in S_{\mathbf{c}}^{[k]}$, for otherwise, the children of u would also belong to $\Delta T_{\mathbf{c}}^{[k]}$, implying that both u and its parent belong $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, in contradiction with (35) and the fact that no state in $S_{\mathbf{c}}^{[k-1]}$ is a proper prefix of another state in $S_{\mathbf{c}}^{[k-1]}$. Thus, we must have

$$\Delta T_{\mathbf{c}}^{[k]} \subseteq S_{\mathbf{c}}^{[k]}\,, \tag{36}$$

which implies that $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq T_{\mathbf{c}}^{[k-1]}$, as claimed.

(ii) Let $r \in \mathcal{A}^*$ and $a \in \mathcal{A}$. By the definition (1), $r \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ if and only if there exists a symbol $b \in \mathcal{A}$ such that $rb \in \Delta T_{\mathbf{c}}^{[k]}$. The claim then follows from the fact that $T_{\mathbf{c}}^{[k]}$ and $T_{\mathbf{c}}^{[k-1]}$ are full trees.

(iii) We first prove (19). The fact that, by Lemma 8, $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$, implies that

$$S_{\mathbf{c}}^{[k-1]} \setminus S_{\mathbf{c}}^{[k]} = S_{\mathbf{c}}^{[k-1]} \cap \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)\,, \tag{37}$$

and the fact that, by (i), $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq T_{\mathbf{c}}^{[k-1]}$, implies that

$$S_{\mathbf{c}}^{[k-1]} \cap \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) = \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)\,. \tag{38}$$

Since $u \in S_{\mathbf{c}}^{[k-1]} \cap \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ implies that the children of u belong to $\Delta T_{\mathbf{c}}^{[k]}$, we have $S_{\mathbf{c}}^{[k-1]} \cap \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, which, together with (35), (37) and (38), prove (19).

It remains to show (18). By the definition (16), we have $S_{\mathbf{c}}^{[k]} \setminus T_{\mathbf{c}}^{[k-1]} \subseteq \Delta T_{\mathbf{c}}^{[k]}$, and by (16) and (36) we also have $\Delta T_{\mathbf{c}}^{[k]} \subseteq S_{\mathbf{c}}^{[k]} \setminus T_{\mathbf{c}}^{[k-1]}$. Thus,

$$\Delta T_{\mathbf{c}}^{[k]} = S_{\mathbf{c}}^{[k]} \setminus T_{\mathbf{c}}^{[k-1]}\,.$$

Also, since $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$, we have $S_{\mathbf{c}}^{[k]} \cap T_{\mathbf{c}}^{[k-1]} = S_{\mathbf{c}}^{[k]} \cap S_{\mathbf{c}}^{[k-1]}$ and, therefore, $S_{\mathbf{c}}^{[k]} \setminus T_{\mathbf{c}}^{[k-1]} = S_{\mathbf{c}}^{[k]} \setminus S_{\mathbf{c}}^{[k-1]}$, which proves (18). ■

Proof of Lemma 11: By Lemma 27, we know that $\mathcal{P}\left(\Delta T^{[k]}\right)$ is a subset of $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ that has empty intersection with $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$. Thus, since, by (18), the elements in $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$ are parents of states of $T_{\mathbf{c}}^{[k]}$, we have $\mathcal{P}\left(\Delta T^{[k]}\right) \subseteq \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ and, moreover, using (19), we conslude that

$$\mathcal{P}\left(\Delta T^{[k]}\right) \subseteq \left(\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right) . \tag{39}$$

Conversely, since $\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, the right-hand side of (39) is a subset of $\left(\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$, which, by (19), equals $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$. Thus, by Lemma 27, we conclude that

$$\left(\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right) \subseteq \mathcal{P}\left(\Delta T^{[k]}\right) ,$$

which completes the proof. ■

APPENDIX B
PROOF OF LEMMA 17

For the proof of Lemma 17 we make use of the following theorem from [23] that we present in a simplified form, which is nevertheless suitable for our setting.

Theorem 8 ([23, Theorem 2]): Suppose $\{M_n\}$ is an ergodic finite-state Markov chain with a set of states S and a stationary distribution π. Let $F : S \to \mathbb{R}$ be any bounded function and $\bar{F} = \sup_s |F(s)|$. Then, for any $\epsilon > 0$, we have

$$P\left\{\sum_{i=0}^{n-1} \left(F(M_i) - \mathrm{E}_\pi\big[\, F \,\big]\right) \geq n\epsilon \;\middle|\; M_0 = m\right\} \leq \exp\left\{-\frac{n-1}{2}\left(\frac{\epsilon}{d\bar{F}} - \frac{3}{n-1}\right)^2\right\} ,$$

provided $n \geq 1 + 3d\bar{F}/\epsilon$, where m is any fixed initial state, d is a positive constant, and $\mathrm{E}_\pi\big[\, F \,\big]$ is the expectation of $F(M_i)$ under the distribution π.

We will rely on Theorem 8 in the proof of the following auxiliary lemma.

Lemma 28: Let T be a tree model such that all conditional probabilities are different from zero. Consider a symbol a and a fixed string w such that $s \prec w$ for some state s of T. Then, there is a constant N such that for every $k \geq 1$ and $n > N$, we have

$$P_T\left\{\left|n_{aw} - \frac{n_{as}}{n_s} n_w\right| \geq k\sqrt{n_w} \;\middle|\; n_w > 0\right\} \leq \frac{4\exp\left\{-Ck^2\right\}}{P_T\left\{n_w > 0\right\}} , \tag{40}$$

where C is a positive constant independent of k.

Proof: Notice that, since all conditional probabilities of T are different from zero, then $\mathrm{P}_T\left\{n_w > 0\right\}$ is positive. The left-hand side of (40) is

$$\mathrm{P}_T\left\{\left|n_{aw} - \frac{n_{as}}{n_s} n_w\right| \geq k\sqrt{n_w} \;\middle|\; n_w > 0\right\} = \mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w} - \frac{n_{as}}{n_s}\right| \geq \frac{k}{\sqrt{n_w}} \;\middle|\; n_w > 0\right\} .$$

Now, recalling that $p_T(a|s)$ denotes the probability of symbol a conditioned on state s, we have

$$\begin{aligned}
&\mathrm{P}_T\left\{\left|n_{aw}-\frac{n_{as}}{n_s}n_w\right|\geq k\sqrt{n_w}\;\middle|\;n_w>0\right\}= \\
&\quad=\quad \mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)+p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{\sqrt{n_w}}\;\middle|\;n_w>0\right\} \\
&\quad\leq\quad \mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)\right|+\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{\sqrt{n_w}}\;\middle|\;n_w>0\right\} \\
&\quad\leq\quad \mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)\right|+\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{\sqrt{n}}\;\middle|\;n_w>0\right\} \qquad (41) \\
&\quad\leq\quad \mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_w>0\right\}+\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_w>0\right\}, \qquad (42)
\end{aligned}$$

where (41) follows from $n\geq n_w$ and (42) from a trivial union bound.

Since $n_w>0$ implies $n_s>0$, the second term on the right-hand side of (42) can be transformed and upper-bounded as follows:

$$\begin{aligned}
\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_w>0\right\} &= \frac{\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}},n_w>0\right\}}{\mathrm{P}_T\left\{n_w>0\right\}} \\
&\leq \frac{\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}},n_s>0\right\}}{\mathrm{P}_T\left\{n_w>0\right\}} \\
&= \frac{\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_s>0\right\}\mathrm{P}_T\left\{n_s>0\right\}}{\mathrm{P}_T\left\{n_w>0\right\}}.
\end{aligned}$$

Substituting in (42), we obtain

$$\begin{aligned}
\mathrm{P}_T\left\{\left|n_{aw}-\frac{n_{as}}{n_s}n_w\right|\geq k\sqrt{n_w}\;\middle|\;n_w>0\right\}\leq\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_w>0\right\}+ \\
+\frac{\mathrm{P}_T\left\{\left|p_T(a|s)-\frac{n_{as}}{n_s}\right|\geq\frac{k}{2\sqrt{n}}\;\middle|\;n_s>0\right\}\mathrm{P}_T\left\{n_s>0\right\}}{\mathrm{P}_T\left\{n_w>0\right\}}. \qquad (43)
\end{aligned}$$

We now bound $\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w}-p_T(a|s)\right|\geq\epsilon\;\middle|\;n_w>0\right\}$ for any fixed string w with $s\preceq w$, and $\epsilon>0$. Notice that this includes the case $w=s$, and, thus, the derived bound will apply to both the first term and the numerator of the second term on the right-hand side of (43).

Let $\{M_i\}$ be a Markov chain defined on the set of states $S=\mathcal{A}^q$, where q is the maximum between $|w|+1$ and the depth of T, with a state transition matrix given by

$$P(M_{i+1}=v|M_i=u)=\begin{cases} p_T(b|t),\ \text{where } b=\text{head}(v) \text{ and } t=\sigma(\overline{u}), & \text{if } \text{tail}(v)\prec u, \\ 0, & \text{otherwise.}\end{cases} \qquad (44)$$

If we let $\{M_i\}$ generate a random sequence $\{X_i\}$ by taking $M_0=x_0x_{-1}\ldots x_{-q+1}$ and $X_i=\text{head}(M_i)$ for all $i>0$, this Markov chain defines exactly the same probability assignment as that defined by T

in (2). Notice that, since $p_T(b|t) > 0$ for all $b \in \mathcal{A}$ and all $t \in S_T$, $\{M_i\}$ is irreducible and aperiodic and, therefore, it has a unique steady state distribution π.

Let $p_w = \sum_{u \in S: w \preceq u} \pi(u)$ and $p_{aw} = \sum_{u \in S: aw \preceq u} \pi(u)$, i.e., the probabilities of emitting the symbol sequences $\overline{w}$ and $\overline{w}a$, respectively, under the steady state distribution π. We have $p_T(a|s) = p_{aw}/p_w$, and therefore

$$\frac{n_{aw}}{n_w} - p_T(a|s) = \frac{n_{aw}/n}{n_w/n} - \frac{p_{aw}}{p_w}\,. \tag{45}$$

Let $0 < \delta < p_w$. When $p_{aw} - \delta < \frac{n_{aw}}{n} < p_{aw} + \delta$, and $p_w - \delta < \frac{n_w}{n} < p_w + \delta$, we have

$$\frac{p_{aw} - \delta}{p_w + \delta} < \frac{n_{aw}/n}{n_w/n} < \frac{p_{aw} + \delta}{p_w - \delta}\,.$$

Thus, the event $\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| < \delta\right\} \cap \left\{\left|\frac{n_w}{n} - p_w\right| < \delta\right\}$ implies

$$\frac{p_{aw}}{p_w} - \left(\frac{p_{aw}}{p_w} - \frac{p_{aw} - \delta}{p_w + \delta}\right) < \frac{n_{aw}/n}{n_w/n} < \frac{p_{aw}}{p_w} + \left(\frac{p_{aw} + \delta}{p_w - \delta} - \frac{p_{aw}}{p_w}\right),$$

or,

$$\frac{p_{aw}}{p_w} - \frac{\delta(p_w + p_{aw})}{p_w(p_w + \delta)} < \frac{n_{aw}/n}{n_w/n} < \frac{p_{aw}}{p_w} + \frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)}\,.$$

Since we have $\frac{\delta(p_w + p_{aw})}{p_w(p_w + \delta)} < \frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)}$, the condition above further implies,

$$\frac{p_{aw}}{p_w} - \frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)} < \frac{n_{aw}/n}{n_w/n} < \frac{p_{aw}}{p_w} + \frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)}\,,$$

which in turn, by (45), yields

$$\left|\frac{n_{aw}}{n_w} - p_T(a|s)\right| < \frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)}\,. \tag{46}$$

Now, given $0 < \epsilon \leq 1$, we take $\delta = \epsilon \frac{p_w^2}{p_w + p_{aw} + 1}$, which satisfies the condition $0 < \delta < p_w$ assumed for the derivation of (46). Then, $\delta \leq \epsilon \frac{p_w^2}{p_w + p_{aw} + \epsilon p_w}$, and therefore $\frac{\delta(p_w + p_{aw})}{p_w(p_w - \delta)} \leq \epsilon$. Thus, from (46) it follows that the event $\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| < \delta\right\} \cap \left\{\left|\frac{n_w}{n} - p_w\right| < \delta\right\}$ implies $\left|\frac{n_{aw}}{n_w} - p_T(a|s)\right| < \epsilon$. Hence, we have

$$\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w} - p_T(a|s)\right| < \epsilon \;\middle|\; n_w > 0\right\} \geq \mathrm{P}_T\left\{\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| < \delta\right\} \cap \left\{\left|\frac{n_w}{n} - p_w\right| < \delta\right\} \;\middle|\; n_w > 0\right\},$$

or,

$$\begin{aligned}
\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w} - p_T(a|s)\right| \geq \epsilon \;\middle|\; n_w > 0\right\} &\leq \mathrm{P}_T\left\{\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| \geq \delta\right\} \cup \left\{\left|\frac{n_w}{n} - p_w\right| \geq \delta\right\} \;\middle|\; n_w > 0\right\} \\
&= \frac{\mathrm{P}_T\left\{\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| \geq \delta\right\} \cup \left\{\left|\frac{n_w}{n} - p_w\right| \geq \delta\right\}, n_w > 0\right\}}{\mathrm{P}_T\left\{n_w > 0\right\}} \\
&\leq \frac{\mathrm{P}_T\left\{\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| \geq \delta\right\} \cup \left\{\left|\frac{n_w}{n} - p_w\right| \geq \delta\right\}\right\}}{\mathrm{P}_T\left\{n_w > 0\right\}} \\
&\leq \frac{\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| \geq \delta\right\}}{\mathrm{P}_T\left\{n_w > 0\right\}} + \frac{\mathrm{P}_T\left\{\left|\frac{n_w}{n} - p_w\right| \geq \delta\right\}}{\mathrm{P}_T\left\{n_w > 0\right\}}\,.
\end{aligned} \tag{47}$$

We now apply Theorem 8 to both terms of the right-hand side of (47). Specifically, for $\{M_i\}$ as in (44) and $M_0 = x_0 x_{-1} \ldots x_{-q+1}$ as before, we define $F(M_i)$ as the indicator function of the condition $aw \preceq M_i$.

Then, we have $\bar{F} = 1$ and, by (4), we also have $n_{aw} = \sum_{i=0}^{n-1} F(M_i)$. Therefore, for the numerator of the first term on the right-hand side of (47) we obtain

$$\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n} - p_{aw}\right| \geq \delta\right\} \leq \exp\left\{-\frac{n-1}{2}\left(\frac{\delta}{d_1} - \frac{3}{n-1}\right)^2\right\}, \tag{48}$$

provided $n \geq 1 + \frac{3d_1}{\delta}$, where $\delta = \epsilon\frac{p_w^2}{p_w+p_{aw}+1}$ as before, and d_1 is a positive constant. Similarly, for the numerator of the second term we obtain

$$\mathrm{P}_T\left\{\left|\frac{n_w}{n} - p_w\right| \geq \delta\right\} \leq \exp\left\{-\frac{n-1}{2}\left(\frac{\delta}{d_2} - \frac{3}{n-1}\right)^2\right\}, \tag{49}$$

provided $n \geq 1 + \frac{3d_2}{\delta}$, where d_2 is a positive constant. Taking $d = \max\{d_1, d_2\}$, and combining (48) and (49) with (47), we obtain

$$\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w} - p_{_T}(a|s)\right| \geq \epsilon \;\middle|\; n_w > 0\right\} \leq \frac{2}{\mathrm{P}_T\{n_w > 0\}}\exp\left\{-\frac{n-1}{2}\left(\frac{\delta}{d} - \frac{3}{n-1}\right)^2\right\}, \tag{50}$$

provided $n \geq 1 + \frac{3d}{\delta}$. Recalling the definition of δ above, we can rewrite (50) as

$$\mathrm{P}_T\left\{\left|\frac{n_{aw}}{n_w} - p_{_T}(a|s)\right| \geq \epsilon \;\middle|\; n_w > 0\right\} \leq \frac{2}{\mathrm{P}_T\{n_w > 0\}}\exp\left\{-\frac{n-1}{2}\left(r\epsilon - \frac{3}{n-1}\right)^2\right\}, \tag{51}$$

provided $n \geq 1 + \frac{r'}{\epsilon}$, where r and r' are positive constants. We now apply the bound in (51) to both the first term and the numerator of the second term on the right hand side of (43) (with $w = s$ for the latter case) and obtain for $1 \leq k \leq 2\sqrt{n}$, with $\epsilon = \frac{k}{2\sqrt{n}}$, for all n satisfying $n \geq 1 + r'\frac{2\sqrt{n}}{k}$,

$$\mathrm{P}_T\left\{\left|n_{aw} - \frac{n_{as}}{n_s}n_w\right| \geq k\sqrt{n_w} \;\middle|\; n_w > 0\right\} \leq \frac{4}{\mathrm{P}_T\{n_w > 0\}}\exp\left\{-\frac{n-1}{2}\left(r\frac{k}{2\sqrt{n}} - \frac{3}{n-1}\right)^2\right\}. \tag{52}$$

Now, there exist positive constants N, C, independent of k, such that, for every $n > N$, the magnitude of the exponent in (52) is bounded as

$$\frac{n-1}{2}\left(r\frac{k}{2\sqrt{n}} - \frac{3}{n-1}\right)^2 \geq Ck^2,$$

and, simultaneously, the condition $n \geq 1 + r'\frac{2\sqrt{n}}{k}$ holds. Thus, (40) is satisfied for all $n > N$ and all k, $1 \leq k \leq 2\sqrt{n}$. On the other hand, for $k > 2\sqrt{n}$, it follows from (42) that the left-hand side of (40) is zero and, therefore, the inequality in (40) is also satisfied in this case. ■

Proof of Lemma 17:

We generalize the definition of m_1 in (23) as

$$m_i = n_u, \quad \text{where } u = t_{\ell-i+1}^q, \quad 0 < i \leq \ell. \tag{53}$$

Thus, m_i counts the number of occurrences in x^n of the suffix $u = t_{\ell-i+1}^q$ of t, and n_i, defined in (23), counts the number of occurrences of the state selected by $\overline{u}$, s_i. Notice that, if $i < j$, then $t_{\ell-i+1}^q$ is a substring of $t_{\ell-j+1}^q$ and, therefore, we have

$$m_i \geq m_j, \quad 0 < i < j \leq \ell. \tag{54}$$

Suppose $\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| < k\sqrt{m_1}$ for some positive number k and all i, $1 < i \leq \ell$. Then, we have

$$\frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1} - k\sqrt{m_1} < m_i < \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1} + k\sqrt{m_1}\,. \tag{55}$$

If $i > 2$, we can apply the same inequalities for m_{i-1}, and combining with (55), we obtain

$$\frac{n_{i-1}^\alpha}{n_{i-1}}\left(\frac{n_{i-2}^\alpha}{n_{i-2}} m_{i-2} - k\sqrt{m_1}\right) - k\sqrt{m_1} < m_i < \frac{n_{i-1}^\alpha}{n_{i-1}}\left(\frac{n_{i-2}^\alpha}{n_{i-2}} m_{i-2} + k\sqrt{m_1}\right) + k\sqrt{m_1}\,,$$

or

$$\frac{n_{i-1}^\alpha}{n_{i-1}}\frac{n_{i-2}^\alpha}{n_{i-2}} m_{i-2} - \left(1 + \frac{n_{i-1}^\alpha}{n_{i-1}}\right) k\sqrt{m_1} < m_i < \frac{n_{i-1}^\alpha}{n_{i-1}}\frac{n_{i-2}^\alpha}{n_{i-2}} m_{i-2} + \left(1 + \frac{n_{i-1}^\alpha}{n_{i-1}}\right) k\sqrt{m_1}\,.$$

Proceeding in the same fashion, starting from $i = \ell$ and iterating down to $i = 2$, we obtain

$$m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} - \left(1 + \sum_{j=2}^{\ell-1} \prod_{i=j}^{\ell-1} \frac{n_i^\alpha}{n_i}\right) k\sqrt{m_1} < m_\ell < m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} + \left(1 + \sum_{j=2}^{\ell-1} \prod_{i=j}^{\ell-1} \frac{n_i^\alpha}{n_i}\right) k\sqrt{m_1}\,.$$

Since $\frac{n_i^\alpha}{n_i} \leq 1$ for all i, we further bound

$$m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} - (\ell-1) k\sqrt{m_1} < m_\ell < m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} + (\ell-1) k\sqrt{m_1}\,.$$

Hence, the event $\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| < k\sqrt{m_1} \text{ for all } i, 1 < i \leq \ell\right\}$ implies that $\left\{\left|m_\ell - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}\right| < k(\ell-1)\sqrt{m_1}\right\}$. Recalling that $Z_t = \left|n_t - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i}\right|$ and, by (53), $m_\ell = n_t$, applying a union bound, we conclude that

$$\mathrm{P}_T\left\{Z_t \geq k(\ell-1)\sqrt{m_1} \;\middle|\; m_{\ell-1} > 0\right\} \leq \sum_{i=2}^{\ell} \mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_1} \;\middle|\; m_{\ell-1} > 0\right\}, \tag{56}$$

where we notice that, by (54), the condition $m_{\ell-1} > 0$ ensures that $m_i > 0$ for all i, $0 < i < \ell$ and, therefore, also $n_i > 0$ for all i, $0 < i < \ell$, so that Z_t is well defined. Moreover, also by (54), for all i, $1 < i \leq \ell$, we have $m_1 \geq m_{i-1}$. Therefore, from (56) we obtain

$$\mathrm{P}_T\left\{Z_t \geq k(\ell-1)\sqrt{m_1} \;\middle|\; m_{\ell-1} > 0\right\} \leq \sum_{i=2}^{\ell} \mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{\ell-1} > 0\right\}. \tag{57}$$

Since $m_{\ell-1} > 0$ implies that $m_{i-1} > 0$ for all i, $1 < i \leq \ell$, each term of the summation in (57) can be transformed and bounded as follows:

$$\begin{aligned}
&\mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{\ell-1} > 0\right\} \\
&\quad = \frac{\mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_{i-1}}, m_{\ell-1} > 0\right\}}{\mathrm{P}_T\left\{m_{\ell-1} > 0\right\}} \\
&\quad \leq \frac{\mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_{i-1}}, m_{i-1} > 0\right\}}{\mathrm{P}_T\left\{m_{\ell-1} > 0\right\}} \\
&\quad = \frac{\mathrm{P}_T\left\{\left|m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1}\right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{i-1} > 0\right\} \mathrm{P}_T\left\{m_{i-1} > 0\right\}}{\mathrm{P}_T\left\{m_{\ell-1} > 0\right\}}.
\end{aligned} \tag{58}$$

Now, with $w = t^q_{\ell-i+2}$ and $a = t_{\ell-i+1}$, we have $m_{i-1} = n_w$ and $m_i = n_{aw}$. Also, recalling the definition of s_i just before (22), for $s = s_{i-1}$, we have $s \prec w$ and, by (23), $n_{i-1} = n_s$ and $n^\alpha_{i-1} = n^{(a)}_s$. Therefore, we have

$$\begin{aligned}\left| m_i - \frac{n^\alpha_{i-1}}{n_{i-1}} m_{i-1} \right| = \left| n_{aw} - \frac{n^{(a)}_s}{n_s} n_w \right| &\leq \left| n_{aw} - \frac{n_{as}}{n_s} n_w \right| + \left| \frac{n_{as}}{n_s} n_w - \frac{n^{(a)}_s}{n_s} n_w \right| \\ &\leq \left| n_{aw} - \frac{n_{as}}{n_s} n_w \right| + 1\,,\end{aligned}$$

where the last inequality follows from the fact that, since $s \prec w$, $n_w/n_s \leq 1$ and, by (7), $|n_{as} - n^{(a)}_s| \leq 1$. As a consequence, we have

$$\begin{aligned}\mathrm{P}_T\left\{ \left| m_i - \frac{n^\alpha_{i-1}}{n_{i-1}} m_{i-1} \right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{i-1} > 0 \right\} &\leq \mathrm{P}_T\left\{ \left| n_{aw} - \frac{n_{as}}{n_s} n_w \right| \geq k\sqrt{n_w} - 1 \;\middle|\; n_w > 0 \right\} \\ &\leq \mathrm{P}_T\left\{ \left| n_{aw} - \frac{n_{as}}{n_s} n_w \right| \geq (k-1)\sqrt{n_w} \;\middle|\; n_w > 0 \right\}.\end{aligned}$$

Thus, by Lemma 28, for $k \geq 2$ we obtain

$$\mathrm{P}_T\left\{ \left| m_i - \frac{n^\alpha_{i-1}}{n_{i-1}} m_{i-1} \right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{i-1} > 0 \right\} \leq \frac{4\exp\left\{-C_i(k-1)^2\right\}}{\mathrm{P}_T\left\{m_{i-1} > 0\right\}}\,, \quad k \geq 2\,,$$

provided $n > N_i$, where N_i and C_i are positive constants independent of k. Replacing in the right-hand side of (58), we obtain

$$\mathrm{P}_T\left\{ \left| m_i - \frac{n^\alpha_{i-1}}{n_{i-1}} m_{i-1} \right| \geq k\sqrt{m_{i-1}} \;\middle|\; m_{\ell-1} > 0 \right\} \leq \frac{4\exp\left\{-C_i(k-1)^2\right\}}{\mathrm{P}_T\left\{m_{\ell-1} > 0\right\}}\,, \quad k \geq 2\,.$$

We now take $C = \min\{C_i : 1 < i \leq \ell\}$, where, since ℓ is no greater than the length of the fixed string t, the minimum is over a finite set and, thus, we have $C > 0$. Then, using (57), we obtain, for n sufficiently large,

$$\mathrm{P}_T\left\{ Z_t \geq k(\ell-1)\sqrt{m_1} \;\middle|\; m_{\ell-1} > 0 \right\} \leq \frac{(\ell-1)4\exp\left\{-C(k-1)^2\right\}}{\mathrm{P}_T\left\{m_{\ell-1} > 0\right\}}\,, \quad k \geq 2\,,$$

or,

$$\mathrm{P}_T\left\{ \frac{Z_t}{\sqrt{m_1}} \geq k(\ell-1) \;\middle|\; m_{\ell-1} > 0 \right\} \mathrm{P}_T\left\{m_{\ell-1} > 0\right\} \leq (\ell-1)4\exp\left\{-C(k-1)^2\right\}\,, \quad k \geq 2\,. \tag{59}$$

Now, the expectation of $\lfloor Z_t \rfloor^\top$ is

$$\mathrm{E}\left[\lfloor Z_t \rfloor^\top \right] = \sum_{i \geq 1} \mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i \right\},$$

or, since, by definition, $\lfloor Z_t \rfloor^\top \leq n$ for sequences of length n,

$$\mathrm{E}\left[\lfloor Z_t \rfloor^\top \right] = \sum_{i=1}^{n+1} \mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i \right\}. \tag{60}$$

We bound each term of (60) as

$$\begin{aligned}\mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i \right\} &\leq \mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i, m_{\ell-1} > 0 \right\} + \mathrm{P}_T\left\{m_{\ell-1} = 0\right\} \\ &= \mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i \;\middle|\; m_{\ell-1} > 0 \right\} \mathrm{P}_T\left\{m_{\ell-1} > 0\right\} + \mathrm{P}_T\left\{m_{\ell-1} = 0\right\}. \end{aligned} \tag{61}$$

By the definition of $\lfloor Z_t \rfloor^\top$, we have

$$\mathrm{P}_T\left\{ \lfloor Z_t \rfloor^\top \geq i \;\middle|\; m_{\ell-1} > 0 \right\} \leq \mathrm{P}_T\left\{ \frac{Z_t}{\sqrt{m_1}} \geq i \;\middle|\; m_{\ell-1} > 0 \right\},$$

so that replacing in (61) and combining with (60), we obtain

$$\mathrm{E}\left[\lfloor Z_t \rfloor^\top \right] \leq \left(\sum_{i=1}^{n+1} \mathrm{P}_T\left\{ \frac{Z_t}{\sqrt{m_1}} \geq i \;\middle|\; m_{\ell-1} > 0 \right\} \mathrm{P}_T\left\{ m_{\ell-1} > 0 \right\} \right) + (n+1)\mathrm{P}_T\left\{ m_{\ell-1} = 0 \right\}. \quad (62)$$

Since $\mathrm{P}_T\left\{ m_{\ell-1} = 0 \right\}$ decays exponentially fast with n and, for n sufficiently large, we can bound all but the first $2(\ell-1)$ terms in the summation of (62) using (59), we conclude that $\mathrm{E}\left[\lfloor Z_t \rfloor^\top \right]$ is upper-bounded by a constant. ∎

APPENDIX C
PROOF OF LEMMA 18

Proof: Consider a string $v \in \mathcal{I}(T_F) \setminus \mathcal{I}(T_{\mathbf{c}})$. Clearly, we must have $v = sw$, where $s \in S_{T_{\mathbf{c}}}$ and $sw \in \mathcal{I}(T_F)$. For $s \in S_{T_{\mathbf{c}}}$, let $W(s) = \{w \in \mathcal{A}^* : sw \in T_F\}$, which can be regarded as a subtree of T_F rooted at s. Thus, we can rewrite the right-hand side of (26) as

$$\begin{aligned} \sum_{v \in \mathcal{I}(T_F) \setminus \mathcal{I}(T_{\mathbf{c}})} (\alpha-1)(\kappa_v - 1) &= \sum_{s \in S_{T_{\mathbf{c}}}} \sum_{w \in \mathcal{I}(W(s))} (\alpha-1)(\kappa_{sw} - 1) \\ &= (\alpha-1) \sum_{s \in S_{T_{\mathbf{c}}}} \left[\left(\sum_{w \in \mathcal{I}(W(s))} \kappa_{sw} \right) - \left| \mathcal{I}(W(s)) \right| \right]. \end{aligned} \quad (63)$$

For $a \in \mathcal{A}$ and $s \in S_{T_{\mathbf{c}}}$, let $W_a(s) = \{w \in \mathcal{A}^* : asw \in T_{\mathbf{c}}\}$. Notice that $W_a(s)$ is either empty, if $as \notin T_{\mathbf{c}}$, or, since $T_{\mathbf{c}}$ is full, $W_a(s)$ can be regarded as a full tree rooted at as. We let $\mathcal{I}(W_a(s))$ be the set of internal nodes of $W_a(s)$, with the convention that $\mathcal{I}(W_a(s))$ is empty if $W_a(s)$ is empty. Then, by (27), we have, for $s \in S_{T_{\mathbf{c}}}$ and $w \in \mathcal{I}(W(s))$,

$$\kappa_{sw} = \alpha - \sum_{a \in \mathcal{A}} \mathbb{1}_{asw \in \mathcal{I}(T_{\mathbf{c}})} = \alpha - \sum_{a \in \mathcal{A}} \mathbb{1}_{w \in \mathcal{I}(W_a(s))}, \quad (64)$$

where $\mathbb{1}_{\mathbf{p}}$ denotes the indicator function of the predicate $\mathbf{p}$. Since T_F is FSM, $W_a(s)$ is a subset of $W(s)$, for otherwise there would exist w such that $asw \in T_{\mathbf{c}} \subseteq T_F$ but $sw \notin T_F$, implying that a suffix of a node in T_F is not in T_F, in contradiction with Theorem 1. Thus, summing (64) over $w \in \mathcal{I}(W(s))$, we obtain

$$\sum_{w \in \mathcal{I}(W(s))} \kappa_{sw} = \alpha \left| \mathcal{I}(W(s)) \right| - \sum_{a \in \mathcal{A}} \left| \mathcal{I}(W_a(s)) \right|,$$

and, replacing in (63),

$$\sum_{v \in \mathcal{I}(T_F) \setminus \mathcal{I}(T_{\mathbf{c}})} (\alpha-1)(\kappa_v - 1) = (\alpha-1) \sum_{s \in S_{T_{\mathbf{c}}}} \left[(\alpha-1) \left| \mathcal{I}(W(s)) \right| - \sum_{a \in \mathcal{A}} \left| \mathcal{I}(W_a(s)) \right| \right]. \quad (65)$$

Let $S_W(s) = W(s) \setminus \mathcal{I}(W(s))$, i.e., the set of leafs of T_F that descend from s. Since $W(s)$ is full, we must have $(\alpha-1)|\mathcal{I}(W(s))| = |S_W(s)| - 1$. Also, clearly, we have $S_{T_F} = \bigcup_{s \in S_{T_{\mathbf{c}}}} S_W(s)$. Hence, we

can rewrite (65) as

$$\sum_{v\in\mathcal{I}(T_F)\setminus\mathcal{I}(T_{\mathbf{c}})}(\alpha-1)(\kappa_v-1) = (\alpha-1)\sum_{s\in S_{T_{\mathbf{c}}}}\left[\left|S_W(s)\right|-1-\sum_{a\in\mathcal{A}}\left|\mathcal{I}\left(W_a(s)\right)\right|\right] \quad (66)$$

$$= (\alpha-1)\left|S_{T_F}\right|-(\alpha-1)\sum_{s\in S_{T_{\mathbf{c}}}}\left[1+\sum_{a\in\mathcal{A}}\left|\mathcal{I}\left(W_a(s)\right)\right|\right]. \quad (67)$$

Now, let $L_a(s)$ be the number of edges of the graph $G_{T_{\mathbf{c}}}=(V_{T_{\mathbf{c}}},E_{T_{\mathbf{c}}})$ associated to the emission of symbol a in state s. Notice that, if $W_a(s)$ is not empty, the emission of symbol a in state s of $T_{\mathbf{c}}$ causes a transition to a state of the form asw, where w is a leaf of $W_a(s)$. If $W_a(s)$ is empty, a determines a unique next state transition from s in $T_{\mathbf{c}}$. Thus, since $W_a(s)$ is either a full tree or the empty set, in which case we have defined $\mathcal{I}\left(W_a(s)\right)$ as empty, we have

$$L_a(s)=(\alpha-1)\left|\mathcal{I}\left(W_a(s)\right)\right|+1\,. \quad (68)$$

Summing (68) over $a\in\mathcal{A}$, we obtain the total number of edges in $E_{T_{\mathbf{c}}}$ departing from s, which is

$$D(s)=\sum_{a\in\mathcal{A}}\left[(\alpha-1)\left|\mathcal{I}\left(W_a(s)\right)\right|+1\right]=\alpha+(\alpha-1)\sum_{a\in\mathcal{A}}\left|\mathcal{I}\left(W_a(s)\right)\right|. \quad (69)$$

Combining (69) with (67), we obtain

$$\sum_{v\in\mathcal{I}(T_F)\setminus\mathcal{I}(T_{\mathbf{c}})}(\alpha-1)(\kappa_v-1) = (\alpha-1)\left|S_{T_F}\right|-\sum_{s\in S_{T_{\mathbf{c}}}}\left[D(s)-1\right]$$

$$= (\alpha-1)\left|S_{T_F}\right|-\left(E_{T_{\mathbf{c}}}-V_{T_{\mathbf{c}}}\right),$$

which, by (25), completes the proof. ■

APPENDIX D
PROOF OF LEMMA 20

We first introduce some notation and show two auxiliary lemmas. We recall the definition of $S_{\mathbf{c}}^{[k]}(r)$ in (20). For $r\in\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, where $h<k\le d$, we let

$$\underline{S}_{\mathbf{c}}^{[k]}(r)=\{\sigma_{\mathbf{c}}^{[k-1]}(\overline{s}):s\in S_{\mathbf{c}}^{[k]}(r)\}\,, \quad (70)$$

where we recall that $\sigma_{\mathbf{c}}^{[k-1]}(\overline{s})$ denotes the state selected by $\overline{s}$ in $T_{\mathbf{c}}^{[k-1]}$, which, by Lemma 8, is well defined. We also define

$$A_k(r) = \{(s,t)\in E_{T_{\mathbf{c}}^{[k]}}:s\in S_{\mathbf{c}}^{[k]}(r)\}\,,$$

$$\underline{A}_k(r) = \{(s,t)\in E_{T_{\mathbf{c}}^{[k-1]}}:s\in \underline{S}_{\mathbf{c}}^{[k]}(r)\}\,,$$

and, for $c\in\mathcal{A}$,

$$A_k^{(c)}(r) = \{(s,t)\in A_k(r):c=\text{head}(t)\}\,, \quad (71)$$

$$\underline{A}_k^{(c)}(r) = \{(s,t)\in \underline{A}_k(r):c=\text{head}(t)\}\,. \quad (72)$$

Lemma 29: Let $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, where $h < k \leq d$. Then,

(i) If $r \notin S_{\mathbf{c}}^{[k-1]}$, we have $\underline{S}_{\mathbf{c}}^{[k]}(r) = S_{\mathbf{c}}^{[k]}(r)$.

(ii) If $r \in S_{\mathbf{c}}^{[k-1]}$, we have $\underline{S}_{\mathbf{c}}^{[k]}(r) = \{r\}$ and $S_{\mathbf{c}}^{[k]}(r) = \{ra : a \in \mathcal{A}\}$. If, in addition, $c \in \mathcal{A}$ is such that $cr \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, then $\underline{A}_k^{(c)}(r) = \{(r, cr)\}$ and $A_k^{(c)}(r) = \{(ra, cra) : a \in \mathcal{A}\}$.

Proof: If $r \notin S_{\mathbf{c}}^{[k-1]}$, since $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ and, by Lemma 10(i), $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq T_{\mathbf{c}}^{[k-1]}$, we must have $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ and, therefore, $\sigma_{\mathbf{c}}^{[k-1]}(\overline{s}) = s$ for all $s \in S_{\mathbf{c}}^{[k]}(r)$. Thus, by (70), we obtain $\underline{S}_{\mathbf{c}}^{[k]}(r) = S_{\mathbf{c}}^{[k]}(r)$.

If $r \in S_{\mathbf{c}}^{[k-1]}$, since $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, then, by Lemma 10(ii)–(iii), all the children of r are states of $T_{\mathbf{c}}^{[k]}$. Hence, the first claim of (ii) follows from the defintions (70) and (20). If also $c \in \mathcal{A}$ is such that $cr \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, then, by Lemma 10(ii)–(iii), $cr \in S_{\mathbf{c}}^{[k-1]}$ and all the children of cr are states of $T_{\mathbf{c}}^{[k]}$. Thus, the second claim follows from (9) and the definitions (72) and (71). ■

Now, we recall that all explicit encoding of counts in RefineTypeClass are done through Procedure P. Thus, in the following lemma, we analyze the number of counts described in an invocation of P(r, c).

Lemma 30: Consider the k-th execution of RefineTypeClass, $h + 1 < k \leq d$, which is invoked from the k-th iteration of the loop in EncodeTypeClass (Figure 2). Then, the number of counts described in an invocation of P(r, c), to encode $N_{s,t}$ for every $s \in S_{\mathbf{c}}^{[k]}(r)$, and every $t \in S_{\mathbf{c}}^{[k]}$ such that $c = \text{head}(t)$, equals $|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)|$.

Proof: When the condition in Step 1 of Procedure P holds true, the procedure describes $\alpha - 1$ counts in Step 2. Since, by the assumptions of Procedure P, $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ and, by the condition in Step 1, $cr \in S_{\mathbf{c}}^{[k-1]}$, then, by (19), we have $cr \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$. Since, also by the condition in Step 1, $r \in S_{\mathbf{c}}^{[k-1]}$, then, by Lemma 29(ii), we conclude that the number of counts described in this case, $\alpha - 1$, coincides with $|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)|$.

If the condition of Step 1 is false, then all explicit descriptions of counts take place in Step 9, where $\alpha - 1$ counts are encoded. Thus, by the loop in Step 6, the condition in Step 7, and the loop in Step 8, we have to prove that, if the condition of Step 1 is false, then

$$|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)| = (\alpha - 1) \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \left|\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)\right| . \tag{73}$$

To prove (73), we will show that

$$|\underline{A}_k^{(c)}(r)| = |U(r)| + \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \left(\left|\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)\right| + |Q(cs)| \right), \tag{74}$$

and

$$|A_k^{(c)}(r)| = |U(r)| + \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \left(\alpha \left|\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)\right| + |Q(cs)| \right), \tag{75}$$

where

$$U(r) = \{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in S_{\mathbf{c}}^{[k]}\},$$

and

$$Q(cs) = \{t \in S_{\mathbf{c}}^{[k]} \cap S_{\mathbf{c}}^{[k-1]} : t = csv, v \in \mathcal{A}^*\}. \tag{76}$$

We start with (74), and we consider first the case in which $cr \in S_{\mathbf{c}}^{[k-1]}$. Since the condition of Step 1 is false, we have $r \notin S_{\mathbf{c}}^{[k-1]}$. Therefore, by Lemma 29(i), we have $\underline{S}_{\mathbf{c}}^{[k]}(r) = S_{\mathbf{c}}^{[k]}(r)$ and, hence, $\underline{A}_k^{(c)}(r)$ is comprised of $|S_{\mathbf{c}}^{[k]}(r)|$ edges, each departing from some $s \in S_{\mathbf{c}}^{[k]}(r)$ and arriving at cr, implying that the left-hand side of (74) equals $|S_{\mathbf{c}}^{[k]}(r)|$. On the other hand, since $cr \in S_{\mathbf{c}}^{[k-1]}$ and, by the assumptions of Procedure P, $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, then, by Lemma 10(ii)–(iii), $cs \in S_{\mathbf{c}}^{[k]}$ for all $s \in S_{\mathbf{c}}^{[k]}(r)$. Hence, we have $|U(r)| = |S_{\mathbf{c}}^{[k]}(r)|$, and the summation in the right-hand side of (74) vanishes, which shows the validity of (74) when $cr \in S_{\mathbf{c}}^{[k-1]}$.

We consider now the case in which $cr \notin S_{\mathbf{c}}^{[k-1]}$. Since, by the assumptions of Procedure P, $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ and, by Lemma 10(i), $\mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \subseteq T_{\mathbf{c}}^{[k-1]}$, the condition $cr \notin S_{\mathbf{c}}^{[k-1]}$ implies that $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$. We claim that

$$|\underline{A}_k^{(c)}(r)| = \sum_{s \in S_{\mathbf{c}}^{[k]}(r)} |\{t \in S_{\mathbf{c}}^{[k-1]} : t = csv, v \in \mathcal{A}^*\}|. \tag{77}$$

If $r \in S_{\mathbf{c}}^{[k-1]}$, then, by Lemma 29(ii), we have $\underline{S}_{\mathbf{c}}^{[k]}(r) = \{r\}$ and, therefore, $\underline{A}_k^{(c)}(r)$ is comprised of as many edges as states of $T_{\mathbf{c}}^{[k-1]}$ that descend from cr, because each of these states is the destination of one and only one edge of $\underline{A}_k^{(c)}(r)$. Equivalently, since $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$,

$$|\underline{A}_k^{(c)}(r)| = \sum_{d \in \mathcal{A}} |\{t \in S_{\mathbf{c}}^{[k-1]} : t = crdv, v \in \mathcal{A}^*\}|. \tag{78}$$

Now, since $r \in S_{\mathbf{c}}^{[k-1]}$ and $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, recalling that $\mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$ and using Lemma 10(ii)–(iii), we conclude that all the children of r belong to $S_{\mathbf{c}}^{[k]}$, implying that $S_{\mathbf{c}}^{[k]}(r) = \{rd : d \in \mathcal{A}\}$. Thus, from (78) we obtain (77). If, instead, $r \notin S_{\mathbf{c}}^{[k-1]}$, then, by Lemma 29(i), we have $\underline{S}_{\mathbf{c}}^{[k]}(r) = S_{\mathbf{c}}^{[k]}(r)$. Hence, since $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$, $\underline{A}_k^{(c)}(r)$ is comprised of as many edges as states of $T_{\mathbf{c}}^{[k-1]}$ that descend from each cs, $s \in S_{\mathbf{c}}^{[k]}(r)$, because each of these states is the destination of one and only one edge of $\underline{A}_k^{(c)}(r)$, proving (77) also in this case.

Having proved (77), we now split the summation therein according to whether $s \in U(r)$. If $s \in U(r)$, i.e., $cs \in S_{\mathbf{c}}^{[k]}$, then, since $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ and, by Lemma 8, $T_{\mathbf{c}}^{[k-1]} \subseteq T_{\mathbf{c}}^{[k]}$, we must also have $cs \in S_{\mathbf{c}}^{[k-1]}$. Therefore, each term corresponding to $s \in U(r)$ in (77) contributes 1 to the sum, and the total contribution of these terms is, thus, $|U(r)|$. If $s \notin U(r)$, then, by (76), we must have $cs \notin S_{\mathbf{c}}^{[k]}$. We also have $cs \in T_{\mathbf{c}}^{[k]}$ because, by the assumptions of Procedure P, the parent r of s satisfies $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$. As a consequence, we must have $cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$. We can then write

$$\begin{aligned} |\underline{A}_k^{(c)}(r)| \;=\; |U(r)| + \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \Big(& |\{t \in S_{\mathbf{c}}^{[k-1]} \setminus S_{\mathbf{c}}^{[k]} : t = csv, v \in \mathcal{A}^*\}| \\ & + |\{t \in S_{\mathbf{c}}^{[k-1]} \cap S_{\mathbf{c}}^{[k]} : t = csv, v \in \mathcal{A}^*\}| \Big), \end{aligned}$$

from which (74) follows by (19) and (76).

Next, we prove (75). Since, by the assumptions of Procedure P, we have $cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, $A_k^{(c)}(r)$ is comprised of as many edges as states of $T_{\mathbf{c}}^{[k]}$ that descend from each cs, $s \in S_{\mathbf{c}}^{[k]}(r)$, because each of these states is the destination of one and only one edge of $A_k^{(c)}(r)$, i.e.,

$$|A_k^{(c)}(r)| = \sum_{s \in S_{\mathbf{c}}^{[k]}(r)} \left|\{t \in S_{\mathbf{c}}^{[k]} : t = csv, v \in \mathcal{A}^*\}\right|. \tag{79}$$

Again, we split the summation in (79) according to whether $s \in U(r)$. If $s \in U(r)$, the set in each addend in (79) is comprised of a single element, so that the partial summation equals $|U(r)|$. If $s \notin U(r)$, we have again $cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$, which implies

$$\begin{aligned} |A_k^{(c)}(r)| \;\; = \;\; |U(r)| + \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \Big(& \left|\{t \in S_{\mathbf{c}}^{[k]} \setminus S_{\mathbf{c}}^{[k-1]} : t = csv, v \in \mathcal{A}^*\}\right| \\ & + \left|\{t \in S_{\mathbf{c}}^{[k]} \cap S_{\mathbf{c}}^{[k-1]} : t = csv, v \in \mathcal{A}^*\}\right| \Big), \end{aligned}$$

or, by (76) and (18),

$$|A_k^{(c)}(r)| = |U(r)| + \sum_{s \in S_{\mathbf{c}}^{[k]}(r) : cs \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \left(\left|\Delta T_{\mathbf{c}}^{[k]}(cs)\right| + |Q(cs)| \right).$$

By Lemma 10(ii) and the definition (21), we have $\left|\Delta T_{\mathbf{c}}^{[k]}(cs)\right| = \alpha \left|\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}(cs)\right)\right|$, from which (75) follows. Subtracting (74) from (75) we obtain (73), which completes the proof of the lemma. ■

Proof of Lemma 20:

We analyze the number of counts described in RefineTypeClass (Figure 3), which implements Step 4 of EncodeTypeClass. RefineTypeClass consists of a main loop in Step 1 that iterates over all $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$. We denote by $C_k(r)$ the number of counts described in the iteration corresponding to $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$.

When $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$, RefineTypeClass takes a symbol d such that $dr \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ in Step 3, and invokes P(r, c) in Step 4 for every $c \in \mathcal{A}_d = \mathcal{A} \setminus \{d\}$. Since, by Lemma 30, each of these invocations describes $|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)|$ counts, the number of counts described in Step 4 is

$$C_k(r) = \sum_{c \in \mathcal{A}_d} \left[|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)| \right]. \tag{80}$$

Now, by Lemma 9 and the definition of the symbol d, we have $dr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right) \setminus \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$ and, therefore, by (19), $dr \in \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$. Since $r \in \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ and, by the definition (17), $\mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right) \subseteq \mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$, then, by (19), we have $r \in S_{\mathbf{c}}^{[k-1]}$. Thus, by Lemma 29(ii), we have $|A_k^{(d)}(r)| - |\underline{A}_k^{(d)}(r)| = \alpha - 1 = |S_{\mathbf{c}}^{[k]}(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|$ and, therefore, (80) can be written as

$$C_k(r) = (|A_k(r)| - |S_{\mathbf{c}}^{[k]}(r)|) - (|\underline{A}_k(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|). \tag{81}$$

We now consider the case in which $r \notin \mathcal{P}\left(\Delta_{\mathbf{c}} T_{\mathbf{c}}^{[k]}\right)$ and $r \in \mathcal{I}\left(T_{\mathbf{c}}^{[k-1]}\right)$, so that the condition in Step 6 is satisfied. We claim that the number of described counts, $C_k(r)$, is given by (81) also in this case. Since all counts are described in Step 10, by Lemma 30 and the loop in Step 9, we need to show that

$$\sum_{c \in \mathcal{A} : cr \in \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)} \left[|A_k^{(c)}(r)| - |\underline{A}_k^{(c)}(r)| \right] = (|A_k(r)| - |S_{\mathbf{c}}^{[k]}(r)|) - (|\underline{A}_k(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|). \tag{82}$$

By Lemma 29(i), we have $S_{\mathbf{c}}^{[k]}(r) = \underline{S}_{\mathbf{c}}^{[k]}(r)$. Therefore, by (82), the claim is proved if we show that $|A_k^{(c)}(r)| = |\underline{A}_k^{(c)}(r)|$ for all $c \in \mathcal{A}$ such that $cr \notin \mathcal{I}\left(T_{\mathbf{c}}^{[k]}\right)$. Indeed, for such $c \in \mathcal{A}$, and $s \in S_{\mathbf{c}}^{[k]}(r)$, $\overline{cs}$ is sufficiently long to determine a state $\sigma_{\mathbf{c}}^{[k]}(\overline{cs})$ in $T_{\mathbf{c}}^{[k]}$ and, a fortiori, it is also sufficiently long to determine a state $\sigma_{\mathbf{c}}^{[k-1]}(\overline{cs})$ in $T_{\mathbf{c}}^{[k-1]}$. Thus, both $A_k^{(c)}(r)$ and $\underline{A}_k^{(c)}(r)$ are comprised of exactly $|S_{\mathbf{c}}^{[k]}(r)|$ edges, one departing from each element of $S_{\mathbf{c}}^{[k]}(r)$.

Finally, we analyze the case in which RefineTypeClass skips to Step 12, which, by the conditions in Steps 2 and 6, and by Lemma 11, occurs if and only if $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$. Since no counts are described in this case, the left-hand side of (81) is zero. We claim that the right-hand side of (81) satisfies

$$(|A_k(r)| - |S_{\mathbf{c}}^{[k]}(r)|) - (|\underline{A}_k(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|) = (\alpha - 1)^2 . \tag{83}$$

Indeed, since $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$, by Lemma 7, r is a state of maximal depth in $T_{\mathbf{c}}^{[k-1]}$ and $S_{\mathbf{c}}^{[k]}(r)$ is comprised of α states of maximal depth in $T_{\mathbf{c}}^{[k]}$. Therefore, we have $|\underline{S}_{\mathbf{c}}^{[k]}(r)| = 1$, $|S_{\mathbf{c}}^{[k]}(r)| = \alpha$, $\underline{A}_k(r)$ is comprised of exactly α edges that depart from r, and $A_k(r)$ is comprised of α^2 edges (exactly α departing from each s in $S_{\mathbf{c}}^{[k]}(r)$).

From the above analysis of cases, we conclude that the number of counts described for each $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$, $C_k(r)$, is given by (81), except when $r \in \mathcal{P}\left(\Delta T^{[k]}\right)$, in which case RefineTypeClass skips to Step 12, where no counts are described and, by (83), the right-hand side of (81) evaluates to $(\alpha - 1)^2$. Thus, over all, the number of counts described by RefineTypeClass is

$$C_k = \left(\sum_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} (|A_k(r)| - |S_{\mathbf{c}}^{[k]}(r)|) - (|\underline{A}_k(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|) \right) - \left|\mathcal{P}\left(\Delta T^{[k]}\right)\right| (\alpha - 1)^2 . \tag{84}$$

By the definition of the truncated trees $T^{[k]}$, a state of $T^{[k-1]}$ is either also a state in $T^{[k]}$, or it is the parent of a full complement of α siblings that are states of $T^{[k]}$. Since, by Lemma 7, $\mathcal{P}\left(\Delta T^{[k]}\right) = S_{T^{[k-1]}} \setminus S_{T^{[k]}}$, we have

$$|S_{T^{[k]}}| = |S_{T^{[k-1]}}| + \left|\mathcal{P}\left(\Delta T^{[k]}\right)\right| (\alpha - 1),$$

and (84) becomes

$$C_k = \left(\sum_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} (|A_k(r)| - |S_{\mathbf{c}}^{[k]}(r)|) - (|\underline{A}_k(r)| - |\underline{S}_{\mathbf{c}}^{[k]}(r)|) \right) - \left(|S_{T^{[k]}}| - |S_{T^{[k-1]}}|\right)(\alpha - 1) . \tag{85}$$

Now, by the definitions (20) and (71), we have

$$V_{T_{\mathbf{c}}^{[k]}} = \bigcup_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} S_{\mathbf{c}}^{[k]}(r) , \qquad E_{T_{\mathbf{c}}^{[k]}} = \bigcup_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} A_k(r) .$$

Also, by (19), a state s in $S_{\mathbf{c}}^{[k-1]}$ is either in $S_{\mathbf{c}}^{[k]}$ or in $\mathcal{P}\left(\Delta T_{\mathbf{c}}^{[k]}\right)$. In the former case, for the parent r of s, we have $s \in \underline{S}_{\mathbf{c}}^{[k]}(r)$. In the latter case, by (18), s is the parent of a state in $S_{\mathbf{c}}^{[k]}$. Therefore, $s \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$ and, by the definition (70), $s \in \underline{S}_{\mathbf{c}}^{[k]}(s)$. In any case, we have $s \in \underline{S}_{\mathbf{c}}^{[k]}(r)$ for some $r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)$. Thus,

again,

$$V_{T_{\mathbf{c}}^{[k-1]}} = \bigcup_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} \underline{S}_{\mathbf{c}}^{[k]}(r)\,, \qquad E_{T_{\mathbf{c}}^{[k-1]}} = \bigcup_{r \in \mathcal{P}\left(S_{\mathbf{c}}^{[k]}\right)} \underline{A}_k(r)\,.$$

Hence, the summation in (85) simplifies to

$$\left(|E_{T_{\mathbf{c}}^{[k]}}| - |V_{T_{\mathbf{c}}^{[k]}}|\right) - \left(|E_{T_{\mathbf{c}}^{[k-1]}}| - |V_{T_{\mathbf{c}}^{[k-1]}}|\right)\,,$$

which completes the proof. ■

APPENDIX E
PROOF OF LEMMA 25

We make use of the following lemma and corollary. Recall that $C(T)$ denotes a penalization function that increases with $|S_T|$.

Lemma 31: Let $\hat{T}$ be a tree estimate for x^n according to (31), and let s and w be strings such that $s \in S_{\hat{T}}$, $s \prec w$, and $n_s > 0$. Then, for any extension T' of $\hat{T}$ that contains w and any $a \in \mathcal{A}$, we have

$$\left| n_w^{(a)} - \frac{n_s^{(a)}}{n_s} n_w \right| \leq \sqrt{2(C(T') - C(\hat{T}))n_s \ln n}\,.$$

Proof:

Since $C(T)$ is increasing in $|S_T|$, it is sufficient to consider the case in which T' is the smallest extension of $\hat{T}(x^n)$ that contains w, i.e., the tree that results from extending $\hat{T}(x^n)$ by adding $w'b$ for all proper prefixes w' of w and all symbols $b \in \mathcal{A}$. Let $W = \{su : su \in S_{T'}\}$. Since $\hat{T}(x^n)$ minimizes the cost function $K(T, x^n)$, we have

$$-\sum_{t \in S_{\hat{T}}, a \in \mathcal{A}} n_t^{(a)} \log \frac{n_t^{(a)}}{n_t} + C(\hat{T}) \log n \leq -\sum_{t \in S_{T'}, a \in \mathcal{A}} n_t^{(a)} \log \frac{n_t^{(a)}}{n_t} + C(T') \log n\,.$$

Therefore,

$$-\sum_{t \in S_{\hat{T}}, a \in \mathcal{A}} n_t^{(a)} \log \frac{n_t^{(a)}}{n_t} + \sum_{t \in S_{T'}, a \in \mathcal{A}} n_t^{(a)} \log \frac{n_t^{(a)}}{n_t} \leq (C(T') - C(\hat{T})) \log n\,,$$

which reduces to

$$-\sum_{a \in \mathcal{A}} n_s^{(a)} \log \frac{n_s^{(a)}}{n_s} + \sum_{su \in W, a \in \mathcal{A}} n_{su}^{(a)} \log \frac{n_{su}^{(a)}}{n_{su}} \leq (C(T') - C(\hat{T})) \log n\,.$$

Now, since $n_s > 0$, we further obtain

$$-\sum_{a \in \mathcal{A}} \frac{n_s^{(a)}}{n_s} \log \frac{n_s^{(a)}}{n_s} + \sum_{su \in W} \frac{n_{su}}{n_s} \sum_{a \in \mathcal{A}} \frac{n_{su}^{(a)}}{n_{su}} \log \frac{n_{su}^{(a)}}{n_{su}} \leq (C(T') - C(\hat{T})) \frac{\log n}{n_s}\,. \tag{86}$$

Let $\hat{p}(\cdot|s)$ be the empirical probability distribution over $\mathcal{A}$ given by $\hat{p}(a|s) = n_s^{(a)}/n_s$ and analogously for $su \in W$, $\hat{p}(a|su) = n_{su}^{(a)}/n_{su}$. Consider also a probability distribution $\hat{p}(\cdot)$ over W given by $\hat{p}(su) = n_{su}/n_s$. Let A, B be random variables such that B takes values in W with $B \sim \hat{p}(\cdot)$, and A takes values

in $\mathcal{A}$ with conditional distribution $P(A = a|B = su) = \hat{p}(a|su)$. Then, the joint distribution of A and B is

$$P(A = a, B = su) = P(A = a|B = su)P(B = su) = \hat{p}(a|su)\hat{p}(su) = \frac{n_{su}^{(a)}}{n_{su}} \frac{n_{su}}{n_s} = \frac{n_{su}^{(a)}}{n_s} ,$$

and, thus, the marginal distribution of A is $A \sim \hat{p}(\cdot|s)$. With these random variables, Equation (86) takes the form

$$I(A; B) = H(A) - H(A|B) \leq (C(T') - C(\hat{T})) \frac{\log n}{n_s} ,$$

where I and H denote the usual mutual information and entropy functions. Let Q be a joint distribution given by the product of the marginal distributions of A, B, i.e.,

$$Q(A = a, B = su) = P(A = a)P(B = su) = \frac{n_s^{(a)}}{n_s} \frac{n_{su}}{n_s} .$$

Then, by Pinsker's inequality [24, Lemma 12.6.1], we have

$$\frac{1}{2 \ln 2} \|P - Q\|_1^2 \leq D(P||Q) = I(A; B) \leq (C(T') - C(\hat{T})) \frac{\log n}{n_s} .$$

Therefore,

$$\left(\sum_{a \in \mathcal{A}, su \in W} |P(a, su) - Q(a, su)| \right)^2 \leq 2(C(T') - C(\hat{T})) \frac{\ln n}{n_s} ,$$

which takes the form

$$\sum_{a \in \mathcal{A}, su \in W} \left| \frac{n_{su}^{(a)}}{n_s} - \frac{n_s^{(a)}}{n_s} \frac{n_{su}}{n_s} \right| \leq \sqrt{2(C(T') - C(\hat{T})) \frac{\ln n}{n_s}} . \tag{87}$$

In particular, taking only the term corresponding to $su = w$ and a specific $a \in \mathcal{A}$ in the summation on the left-hand side of (87), we conclude that, for any $a \in \mathcal{A}$,

$$\left| n_w^{(a)} - \frac{n_s^{(a)}}{n_s} n_w \right| \leq \sqrt{2(C(T') - C(\hat{T})) n_s \ln n} . \qquad \blacksquare$$

With a linear penalization function of the form $C(T) = \beta |S_T|$, Lemma 31 yields the following corollary.

Corollary 4: Let $\hat{T}$ be a tree estimate for x^n according to (31) with a penalization function $C(T) = \beta |S_T|$, $s \in S_{\hat{T}}$, $s \prec w$, and $n_s > 0$. Then, for every $a \in \mathcal{A}$,

$$\left| n_w^{(a)} - \frac{n_s^{(a)}}{n_s} n_w \right| \leq \sqrt{2\beta\alpha |w| n_s \ln n} .$$

Proof: Let T' be the smallest extension of $\hat{T}$ that contains w. Thus, $|S_{T'}| - |S_{\hat{T}}| = (\alpha - 1)(|w| - |s|) + 1$ and, since $|w| - |s| \geq 1$ for $s \prec w$, we obtain $|S_{T'}| - |S_{\hat{T}}| \leq \alpha(|w| - |s|) \leq \alpha |w|$. Since the penalization function is linear, we have $C(T') - C(\hat{T}) = \beta(|S_{T'}| - |S_{\hat{T}}|)$ and the claim follows from Lemma 31. $\blacksquare$

Proof of Lemma 25: Let i be an index in the range $1 < i \leq \ell$, and let $w = t_{\ell - i + 2}^q$ and $a = t_{\ell - i + 1}$. Then we have $m_{i-1} = n_w$ and $m_i = n_{aw}$. Also, recalling the definition of s_i just before (22), for $s = s_{i-1}$, we have $s \prec w$ and, by (23), $n_{i-1} = n_s$ and $n_{i-1}^{\alpha} = n_s^{(a)}$. Thus,

$$\left| m_i - \frac{n_{i-1}^{\alpha}}{n_{i-1}} m_{i-1} \right| = \left| n_{aw} - \frac{n_s^{(a)}}{n_s} n_w \right| \leq \left| n_w^{(a)} - \frac{n_s^{(a)}}{n_s} n_w \right| + 1 ,$$

where the inequality follows from the fact that, by (7), $|n_{aw} - n_w^{(a)}| \leq 1$. Thus, by Corollary 4, using the fact that $|w| \leq |t|$, we obtain

$$\frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1} - \sqrt{2\beta\alpha|t|n_s \ln n} - 1 \leq m_i \leq \frac{n_{i-1}^\alpha}{n_{i-1}} m_{i-1} + \sqrt{2\beta\alpha|t|n_s \ln n} + 1 \,. \tag{88}$$

As in the proof of Lemma 17, if $i > 2$, we apply the same inequalities for m_{i-1}, obtaining

$$\left| m_i - \frac{n_{i-1}^\alpha}{n_{i-1}} \frac{n_{i-2}^\alpha}{n_{i-2}} m_{i-1} \right| \leq \left(1 + \frac{n_{i-1}^\alpha}{n_{i-1}} \right) \left(\sqrt{2\beta\alpha|t|n_s \ln n} + 1 \right) .$$

Proceeding in the same fashion, starting from $i = \ell$ and iterating down to $i = 2$, we obtain

$$\left| m_\ell - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} \right| \leq \left(1 + \sum_{j=2}^{\ell-1} \prod_{i=j}^{\ell-1} \frac{n_i^\alpha}{n_i} \right) \left(\sqrt{2\beta\alpha|t|n_s \ln n} + 1 \right) .$$

Since $\frac{n_i^\alpha}{n_i} \leq 1$ for all i, $1 < i < \ell$, we can write

$$\left| m_\ell - m_1 \prod_{i=1}^{\ell-1} \frac{n_i^\alpha}{n_i} \right| \leq (\ell - 1) \left(\sqrt{2\beta\alpha|t|n_s \ln n} + 1 \right) ,$$

from which (32) follows recalling that $m_\ell = n_t$. ■

REFERENCES

[1] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. New York: Academic, 1981.

[2] I. Csiszár, "The method of types," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2505–2523, Oct. 1998.

[3] Y. M. Shtarkov, "Universal sequential coding of single messages," *Problems of Inform. Trans.*, vol. 23, pp. 175–186, Jul. 1987.

[4] P. Myllymäki, "Recent advances in computing the NML for discrete Bayesian networks," in *Proceedings of the First Workshop on Information Theoretic Methods in Science and Engineering*, Tampere, Finland, 2008.

[5] R. E. Krichevskii and V. K. Trofimov, "The performance of universal encoding," *IEEE Trans. Inform. Theory*, vol. 27, pp. 199–207, Mar 1981.

[6] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Trans. Inform. Theory*, vol. 30, pp. 629–636, Jul. 1984.

[7] L. D. Brown, *Fundamentals of statistical exponential families with applications in statistical decision theory*. Hayward, CA: Institute of Mathematical Statistics, 1986.

[8] N. Merhav and M. J. Weinberger, "On universal simulation of information sources using training data," *IEEE Trans. Inform. Theory*, vol. 50, Jan. 2004.

[9] R. B. Ash, *Information Theory*. Wiley, 1967.

[10] V. T. Stefanov, "Noncurved exponential families associated with observations over finite-state Markov chains," *Scand. J. Statist*, vol. 18, pp. 353–356, 1991.

[11] M. J. Weinberger, N. Merhav, and M. Feder, "Optimal sequential probability assignment for individual sequences," *IEEE Trans. Inform. Theory*, vol. 40, pp. 384–396, Mar. 1994.

[12] J. Rissanen, "Complexity of strings in the class of Markov sources," *IEEE Trans. Inform. Theory*, vol. 32, pp. 526–532, Jul. 1986.

[13] M. J. Weinberger, J. Rissanen, and M. Feder, "A universal finite memory source," *IEEE Trans. Inform. Theory*, vol. 41, pp. 643–652, May 1995.

[14] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inform. Theory*, vol. 41, pp. 653–664, May 1995.

[15] J. Takeuchi and T. Kawabata, "Exponential curvature of Markov models," in *Proc. 2007 International Symposium on Information Theory*, Nice, France, Jun. 2007.

[16] Á. Martín, G. Seroussi, and M. J. Weinberger, "Linear time universal coding and time reversal of tree sources via fsm closure," *IEEE Trans. Inform. Theory*, vol. 50, no. 7, pp. 1442–1468, Jul. 2004.

[17] P. L. Buhlmann and A. Wyner, "Variable length Markov chains," *Annals of Statistics*, vol. 27, pp. 480–513, 1998.

[18] Á. Martín, G. Seroussi, and M. J. Weinberger, "Type classes of context trees," submitted. A preliminary version appeared in the Proceedings of the 2007 International Symposium on Information Theory (ISIT'07), Nice, France, 2007.

[19] S. W. Golomb, "Run-length encodings," *IEEE Trans. Inform. Theory*, vol. 12, pp. 399–401, Jul. 1966.

[20] J. Rissanen and G. G. Langdon, "Universal modeling and coding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 12–23, Jan. 1981.

[21] I. Csiszár and Z. Talata, "Context tree estimation for not necessarily finite memory processes, via BIC and MDL," *IEEE Trans. Inform. Theory*, vol. 52, no. 3, pp. 1007–1016, March 2006.

[22] R. Nohre, "Some topics in descriptive complexity," Ph.D. dissertation, Department of Computer Science, The Technical University of Linkoping, Sweden, 1994.

[23] I. Kontoyiannis, L. Lastras-Montano, and S. Meyn, "Relative entropy and exponential deviation bounds for general Markov chains," in *Proc. 2005 International Symposium on Information Theory*, Adelaide, Australia, Sep. 2005, pp. 1563– 1567.

[24] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.