On universal types

Gadiel Seroussi
Information Theory Research
HP Laboratories Palo Alto
HPL-2004-153
September 10, 2004*

E-mail: seroussi@hpl.hp.com

types, type classes, Lempel-Ziv coding, universal simulation, random number generation

We define the universal type class of a sequence x^n, in analogy to the notion used in the classical method of types. Two sequences of the same length are said to be of the same universal (LZ) type if and only if they yield the same set of phrases in the incremental parsing of Ziv and Lempel (1978). We show that the empirical probability distributions of any finite order of two sequences of the same universal type converge, in the variational sense, as the sequence length increases. Consequently, the normalized logarithms of the probabilities assigned by any kth order probability assignment to two sequences of the same universal type, as well as the kth order empirical entropies of the sequences, converge for all k. We study the size of a universal type class, and show that its asymptotic behavior parallels that of the conventional counterpart, with the LZ78 code length playing the role of the empirical entropy. We also estimate the number of universal types for sequences of length n, and show that it is of the form $\exp((1+o(1))\gamma n/\log n)$ for a well characterized constant γ. We describe algorithms for enumerating the sequences in a universal type class, and for drawing a sequence from the class with uniform probability. As an application, we consider the problem of universal simulation of individual sequences. A sequence drawn with uniform probability from the universal type class of x^n is an optimal simulation of x^n in a well defined mathematical sense.

* Internal Accession Date Only
Approved for External Publication

On universal types

Gadiel Seroussi*

Abstract

We define the universal type class of a sequence x^n, in analogy to the notion used in the classical method of types. Two sequences of the same length are said to be of the same universal (LZ) type if and only if they yield the same set of phrases in the incremental parsing of Ziv and Lempel (1978). We show that the empirical probability distributions of any finite order of two sequences of the same universal type converge, in the variational sense, as the sequence length increases. Consequently, the normalized logarithms of the probabilities assigned by any kth order probability assignment to two sequences of the same universal type, as well as the kth order empirical entropies of the sequences, converge for all k. We study the size of a universal type class, and show that its asymptotic behavior parallels that of the conventional counterpart, with the LZ78 code length playing the role of the empirical entropy. We also estimate the number of universal types for sequences of length n, and show that it is of the form $\exp((1+o(1))\gamma n/\log n)$ for a well characterized constant γ. We describe algorithms for enumerating the sequences in a universal type class, and for drawing a sequence from the class with uniform probability. As an application, we consider the problem of universal simulation of individual sequences. A sequence drawn with uniform probability from the universal type class of x^n is an optimal simulation of x^n in a well defined mathematical sense.

Index Terms—method of types, type classes, Lempel-Ziv coding, universal simulation, random process simulation

1 Introduction

Let $\mathcal{A}$ be a finite alphabet of cardinality $\alpha = |\mathcal{A}| \geq 2$. We denote by x_j^k the sequence $x_j x_{j+1} \ldots x_k$, $x_i \in \mathcal{A}$, $j \leq i \leq k$, with the subscript j sometimes omitted from x_j^k when $j = 1$. If $j > k$, $x_j^k = \lambda$, the null string. The terms "string" and "sequence" are used interchangeably; we denote by $\mathcal{A}^*$ (resp. $\mathcal{A}^n$) the set of finite strings (resp. strings of length n) over $\mathcal{A}$, by vw the concatenation of $v, w \in \mathcal{A}^*$, and by $|w|$ the length of a string $w \in \mathcal{A}^*$ (the distinction from set cardinality being clear from the context).

The *method of types* [1, 2] has been very fruitful in several areas of information theory, including source and channel coding (cf. [1], [3, Ch. 12], and [2] for examples). Although often discussed for the case of general discrete memoryless distributions, the method applies to wider classes of parametric probability distributions on sequences over discrete alphabets. Specifically, consider a class $\mathcal{P}$ of probability distributions $P_{\boldsymbol{\Theta}}$ on $\mathcal{A}^n$, $n \geq 1$, parametrized by a finite-dimensional vector

*Hewlett-Packard Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304, USA. E-mail: seroussi@hpl.hp.com.

$\boldsymbol{\Theta}$ of real-valued parameters taking values on $\mathcal{D} \subseteq \mathbb{R}^K$. The *type class* of x^n with respect to $\mathcal{P}$ is defined [2, Sec. VII] as the set

$$\mathcal{T}_{x^n}^{\mathcal{P}} = \{\, y^n \in \mathcal{A}^n \,:\, P_{\boldsymbol{\Theta}}(x^n) = P_{\boldsymbol{\Theta}}(y^n),\ \forall \boldsymbol{\Theta} \in \mathcal{D} \,\}.$$

Generally, type classes are characterized by a set of minimal sufficient statistics[1] (cf. [3, Ch. 2]), whose structure is determined by the class $\mathcal{P}$. For example, in the case where the components of x^n are independent and identically distributed (i.i.d.), and the class $\mathcal{P}$ is parametrized by the $\alpha - 1$ free parameters corresponding to the probabilities of individual symbols from $\mathcal{A}$, the type class of x^n consists of all sequences that have the same single-symbol empirical distribution as x^n [1]. Type classes for families of memoryless distributions with more elaborate parametrizations are discussed in [5]. In the case of finite memory (Markov) distributions of a given order k (with appropriate assumptions on the initial conditions), the type classes are determined by empirical joint distributions of order $k + 1$.

In all the cases mentioned, to define the type classes, one needs knowledge on the structure (e.g., number and nature of the parameters) of $\mathcal{P}$. In this paper, we define a notion of *universal type* that is not explicitly based on such knowledge. The universal type class of x^n will be characterized, as in the conventional case, by the combinatorial structure of x^n. Rather than explicit symbol counts, however, we will base the characterization on the data structure built by a universal data compression scheme, namely, the variant of Lempel-Ziv compression described in [6], often referred to as LZ78.[2]

The *incremental parsing rule* [6] parses the string x^n as $x^n = \mathbf{p}_0\mathbf{p}_1\mathbf{p}_2 \dots \mathbf{p}_{c-1}\mathbf{t}_x$, where $\mathbf{p}_0 = \lambda$, and the *phrase* $\mathbf{p}_i$, $1 \le i < c$, is the shortest substring of x^n starting at the point following $\mathbf{p}_{i-1}$ such that $\mathbf{p}_i \neq \mathbf{p}_j$ for all $j < i$ (x_1 is assumed to follow $\mathbf{p}_0$, which will also be counted as a phrase). The substring $\mathbf{t}_x$, referred to as the *tail* of x^n, is a (possibly empty) suffix for which the parsing rule was truncated due to the end of the string x^n. Conversely, we refer to the prefix $\mathbf{p}_1\mathbf{p}_2 \dots \mathbf{p}_{c-1}$ as the *head* of x^n. Notice that, by construction, all the phrases are distinct, every phrase except λ is an extension by one symbol of another phrase, and every prefix of a phrase is also a phrase. Also, $\mathbf{t}_x$ must be equal to one of the phrases, for otherwise an additional phrase could have been parsed. The number of phrases is a function, $c(x^n)$, of the input sequence.

Let $T_{x^n} = \{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{c-1}\}$ denote the set of phrases, or *dictionary*, in the incremental parsing of x^n. We define the *universal (LZ) type class* (in short, UTC) of x^n, denoted $\mathcal{U}_{x^n}$, as the set

$$\mathcal{U}_{x^n} = \{\, y^n \in \mathcal{A}^n \,:\, T_{y^n} = T_{x^n} \,\}.$$

For arbitrary strings u^k, v^m, $m \ge k \ge 1$, let

$$N(u^k, v^m) = |\{\, i \,:\, v_i^{i+k-1} = u^k,\ \ 1 \le i \le m - k + 1 \,\}|$$

[1] For concreteness, when referring to *conventional* types, we restrict our attention to *exponential families* of distributions [4], which include many of the parametric families of interest in information theory, and provide an appropriate frame of reference for universal types.

[2] Similar notions of universal type can be defined also for other universal compression schemes, e.g. Context [7]. Presently, however, the LZ78 scheme appears more amenable to a combinatorial characterization of its type classes.

denote the number of (possibly overlapping) occurrences of u^k in v^m. Denote the empirical (joint) distribution of order k, $1 \le k \le n$, of x^n by $\hat{P}_{x^n}^{(k)}$, with $\hat{P}_{x^n}^{(k)}(u^k) = N(u^k, x^n)/(n-k+1)$, $u^k \in \mathcal{A}^k$.

For two distributions P and Q on some finite alphabet Ω, the L_∞ distance between P and Q is defined, as is customary, as

$$\|P - Q\|_\infty = \max_{\omega \in \Omega} |P(\omega) - Q(\omega)|.$$

(Since we work with finite alphabets, all convergence statements in the paper where this metric is used will essentially apply also to the variational distance $\|P - Q\|_1 = \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)|$.)

A fundamental property of the UTC is given in the following theorem, which we prove in Section 2. In the theorem, and throughout the paper, we use the notation $\{x^n\}$ for a sequence of sequences x^n, indexed by the sequence length n, where the latter is assumed to take on an infinite, increasing (and possibly sparse) sequence of positive integer values. The setting includes, but is not limited to, the situation where the finite sequences x^n are prefixes of one common semi-infinite sequence x^∞. In general, we assume no consistency between the sequences x^n for different values of n. For succinctness, in the sequel we still refer to "a sequence" $\{x^n\}$, with this interpretation understood.

Theorem 1 *Let $\{x^n\}$ be an arbitrary sequence, and k a positive integer. If $y^n \in \mathcal{U}_{x^n}$, then,*

$$\lim_{n \to \infty} \|\hat{P}_{x^n}^{(k)} - \hat{P}_{y^n}^{(k)}\|_\infty = 0, \tag{1}$$

with convergence rate $O(1/\log n)$, uniformly in x^n and y^n.

A kth order (*finite-memory*, or *Markov*) probability assignment $Q^{(k)}$ is defined by a set of conditional probability distributions $Q^{(k)}(u_{k+1}|u_1^k)$, $u^{k+1} \in \mathcal{A}^{k+1}$, and a distribution $Q^{(k)}(x_1^k)$ on the initial state, so that $Q^{(k)}(x_1^n) = Q^{(k)}(x_1^k) \prod_{i=k+1}^n Q^{(k)}(x_i|x_{i-k}^{i-1})$. In particular, $Q^{(k)}$ could be defined by the kth order approximation of an ergodic measure [3]. The following is an immediate consequence of Theorem 1.

Corollary 1 *Let $\{x^n\}$ and $\{y^n\}$ be sequences such that $y^n \in \mathcal{U}_{x^n}$. Then, for any nonnegative integer k, and any kth order probability assignment $Q^{(k)}$ such that $Q^{(k)}(x^n) \neq 0$ and $Q^{(k)}(y^n) \neq 0$, we have*

$$\lim_{n \to \infty} \left| \frac{1}{n} \log \frac{Q^{(k)}(x^n)}{Q^{(k)}(y^n)} \right| = 0,$$

with convergence rate $O(1/\log n)$, uniformly in x^n and y^n.

Theorem 1 and Corollary 1 are universal analogues of the defining properties of conventional types. In a conventional type class, all the sequences in the class have the *same* statistics relative to the model class defining the types (e.g., k+1-st order joint empirical distributions for kth order finite-memory), and they are assigned *identical* probabilities by any distribution from the model class. In a sense, both properties mean that sequences from the same type class are statistically "indistinguishable" by distributions in the model class. In the universal type case, "same empirical

distribution" and "identical probabilities" are weakened to asymptotic notions, i.e. "equal in the limit," but they hold for *any* model order. The weakened "indistinguishability" is the price paid for universality.

In addition to the basic statistical properties advanced in Theorem 1 and Corollary 1, we ask about universal types some of the same basic questions that are asked in the study of conventional types. These include questions about the size of the universal type classes, the number of universal types, how to enumerate sequences in type classes and draw random sequences uniformly from them, and how to use type classes in enumerative coding. While answering these questions, we observe that the parallels pointed out in Theorem 1 and Corollary 1 can be established also for the other properties.

We also present an application of universal types to simulation of individual sequences. Simulation of random variables and processes has received significant attention in the literature (see, e.g., [8, 9, 10, 11, 5] and references therein), given its various applications, which include speech and image synthesis, texture reproduction, and simulation of communication systems, to cite a few. In our universal setting, given an individual sequence x^n, we wish to produce a (possibly random) sequence y^n satisfying two requirements: (S1) y^n is statistically "similar" to x^n, and (S2) given that y^n satisfies S1, there is as much uncertainty as possible in the choice of y^n. The problem is similar in flavor to that studied in [5], except that we do not make any stochastic assumptions on x^n, and, in particular, we do not assume it has been emitted by a probabilistic source. After formalizing the requirements (S1) and (S2), we prove that a simulation scheme $\mathcal{S}_{\mathcal{U}}$, based on universal types, satisfies (S1), and is optimal in the sense that no scheme that satisfies (S1) can do significantly better than $\mathcal{S}_{\mathcal{U}}$ with respect to (S2).

We note that the universal types defined in this paper are conceptually related to the countably-parametrized types studied in [12] for renewal and related processes. The types of [12] can also be regarded as being defined by a parsing of x^n, where phrases are runs of '0's ended by delimiting '1's. Two sequences are of the same type if and only if they have the same multi-set of phrases. Conversely, in the case of our universal types, the LZ dictionary T_{x^n} could be regarded as a countable set of statistics. We also note the sets of sequences obtained from a given sequence x^n by permuting phrases of fixed length, constrained by the state transitions of a finite-state machine, were studied in [13, 14], where they are referred to as "conditional types."

The rest of the paper is organized as follows. In Section 2 we define the basic combinatorial tools that will be used throughout the work, and recall some of the main properties of the LZ78 incremental parsing. With this background established, we prove Theorem 1 and Corollary 1, and present some additional properties of universal types. In Section 3 we study the number of sequences in a universal type class. We first give an exact combinatorial formula, and then study its asymptotic properties. We show that the size of a universal type class behaves like that of a conventional type class, with the LZ78 normalized code length playing the role of the empirical entropy rate. In Section 4 we estimate the number of universal types for sequences of a given length n. We show that the problem is equivalent to a basic question in the combinatorics of α-ary trees,

which had been hitherto unanswered. A sharp estimate was proved in [15] (published independently due to its intrinsic combinatorial interest); we present the statement of the main result of [15] here, together with its consequences on the number of universal types. In particular, we show that the number of universal types is sub-exponential in n, as generally desired. In Section 5, we present algorithms for enumerating the sequences in a universal type class, and for selecting a sequence with uniform probability from the class. In Section 6 we apply these algorithms to define the universal simulation scheme for individual sequences mentioned above. We also present an example of simulation of graphic textures.

Throughout the paper $\exp_b(x)$ denotes b^x, $\log x = \log_2 x$, and $\ln x = \log_e x$. Entropies will be measured in bits and denoted by an italicized H; entropy rates will be measured in bits per α-ary symbol and denoted by a bold $\mathbf{H}$.

2 The universal type class of x^n

2.1 Parsing trees

We identify, without loss of generality, the alphabet $\mathcal{A}$ with the set $\{0, 1, \ldots, \alpha-1\}$. A *$\alpha$-ary tree* T is recursively defined as either being an empty set, or consisting of a *root node* and the nodes of α disjoint, ordered α-ary trees $T^{(0)}, T^{(1)}, \ldots, T^{(\alpha-1)}$, any number of which may be empty [16]. For $a \in \mathcal{A}$, if $T^{(a)}$ is not empty, we say that there is a *branch* labeled with the symbol a, going from the root of T to the root of $T^{(a)}$; the latter is called a *child* of the root of T. The number of children of a node v (its *(out-)degree*) will be denoted $\deg(v)$. A node v with $\deg(v) = 0$ is called a *leaf*, otherwise, v is an *internal* node. An α-ary tree is *full* if $deg(v) = \alpha$ for every internal node v. The *depth* of a node is the number of branches on the path from the root to the node. We will make free use of conventional tree terminology derived naturally from the child relation, referring, for example, to the *parent, sibling, ancestors*, and *descendants* of a node (a node is considered both an ancestor and a descendant of itself). In graphical representations, we adopt the convention that the root of a tree is at its top and the leaves at its bottom. All trees in the paper will be assumed α-ary, and will be referred to simply as "trees," unless we wish to emphasize a specific alphabet size (e.g., "binary tree").

The set of phrases in the incremental parsing of x^n is conveniently represented by a tree, which is constructed by starting from a root node, and adding, for every phrase $\mathbf{p} = a_1 a_2 \ldots a_m \in T_{x^n}$, a path with branches labeled $a_1, a_2, \ldots, a_m$ from the root of the tree. By the properties of the incremental parsing, there is a one-to-one correspondence between the nodes of a tree thus constructed and the phrases in T_{x^n}, with the root corresponding to $\mathbf{p}_0 = \lambda$. We call this tree the *parsing tree* of x^n, and, since it completely characterizes the set of phrases, we denote it also by T_{x^n}. We will refer to phrases and nodes indistinctly, and will use set notation for nodes in trees (writing, for example, $v \in T$ when v is a node or phrase in a tree T). The number of nodes in the parsing tree of x^n is $c(x^n)$, and its *path length* [16] is

$$\tilde{n} = |\mathbf{p}_1| + |\mathbf{p}_2| + \cdots + |\mathbf{p}_{c-1}| = n - |\mathbf{t}_x|. \tag{2}$$

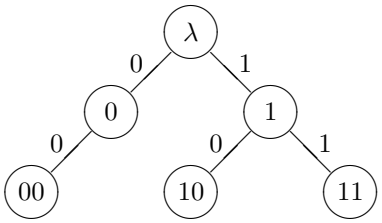

Figure 1: Parsing tree for the binary string $x^8 = 10101100$

The *height* of T_{x^n}, denoted $h(T_{x^n})$, is the maximum depth of a node, or, equivalently, the length of the longest phrase in the parsing. The number of leaves in T_{x^n} will be denoted $\ell(T_{x^n})$. Global tree parameters such as c, h, ℓ, and $\tilde{n}$, will be referred to as *parsing parameters*. We loosely allow either x^n or T_{x^n} to act as the argument of the function associated with a parsing parameter, and in any case we omit that argument when clear from the context.

All the sequences in a UTC share the same parsing tree T, which can serve as a canonical representation of the type class. In general, a complete specification of the UTC requires also the sequence length n, since the same parsing tree T might result from parsing sequences of different lengths, due to possibly different tail lengths. For a given tree T, n can vary in the range $\tilde{n} \leq n \leq \tilde{n} + h$, where h is the height of T. When $\tilde{n} = n$ (i.e., x^n has a null tail), we say that $\mathcal{U}_{x^n}$ is the *natural* UTC associated with T_{x^n}. If T is an arbitrary parsing tree, we denote the associated natural UTC by $\mathcal{U}(T)$, without reference to a specific string in the type class. We call a valid pair (T, n), the *universal type* (UT) of the sequences in the corresponding UTC. When n is not specified, the natural UT is assumed. Clearly, the head of x^n is $x^{\tilde{n}}$, and $\mathcal{U}(T_{x^n}) = \mathcal{U}_{x^{\tilde{n}}}$.

Example 1. Consider the binary string $x^8 = 10101100$, with $n = 8$. The incremental parsing for x^8 is $\lambda, 1, 0, 10, 11, 00$, with $c = 6$ and a null tail. Therefore, $\tilde{n} = n = 8$. The corresponding parsing tree is shown in Figure 1. The sequence $y^8 = 01001011$ is parsed into $\lambda, 0, 1, 00, 10, 11$, defining the same set of phrases as x^8. Thus, x^8 and y^8 are of the same (natural) universal type. □

The following lemma presents some well known properties of the incremental parsing, which derive in a straightforward manner from basic properties of α-ary trees (see, e.g., [16]).

Lemma 1 *Let $\{x^n\}$ be an arbitrary sequence with $c = c(x^n)$. The following relations hold.*

(i) $c \log_\alpha c - \nu c + O(\log c) \leq n \leq c(c-1)/2$, *where* $\nu = \alpha/(\alpha-1) - \log_\alpha(\alpha-1)$.

(ii) $\left\lceil \sqrt{2n} \right\rceil \leq c \leq \dfrac{n}{\log_\alpha n - O(\log\log n)}$.

(iii) $h(x^n) = \max\limits_{1 \leq i \leq c} |\mathbf{p}_i| \leq \left\lfloor \sqrt{2n} \right\rfloor$.

Remarks. A lower bound $c \log_\alpha c - O(c)$ on n as in Lemma 1(i) (or corresponding upper bound on c in (ii)) are attained when T_{x^n} is a *complete* tree with c nodes [16]. In such a tree, all the

leaves are either at depth h or at depth $h-1$.[3] The bound is tight with constant ν when T_{x^n} is also *perfectly balanced*, i.e., when $c = (\alpha^{h+1}-1)/(\alpha-1)$ (the lower bound with ν still holds for arbitrary c). The upper bound on n in Lemma 1(i) (and corresponding lower bound on c in (ii)) are attained when T_{x^n} has a "linear" structure, with all nodes having degree one, except a unique leaf. Such a tree is obtained when parsing a sequence of the form $x^n = aa\ldots a$, for some $a \in \mathcal{A}$, and it also attains the upper bound on height in (iii).

The incremental parsing is best known as the core of the LZ78 universal data compression scheme of Ziv and Lempel [6]. The code length assigned by the LZ78 scheme is characterized in the following lemma.[4]

Lemma 2 ([6]) *The code length assigned by the LZ78 scheme to x^n is*

$$\mathcal{L}_{LZ}(x^n) = c\log c + O(c) \quad \textit{bits.}$$

It is well known (cf. [6, 3]) that the normalized LZ78 code length, $\overline{\mathcal{L}}_{\mathrm{LZ}}(x^n) = \mathcal{L}_{\mathrm{LZ}}(x^n)/n$ does not exceed, in the limit, the code length assigned by any finite-state encoder for the individual sequence x^n; when x^n is emitted by a stationary ergodic source, $\overline{\mathcal{L}}_{\mathrm{LZ}}(x^n)$ converges almost surely to the entropy rate of the source.

2.2 Convergence of statistics

We present proofs for Theorem 1 and Corollary 1.

Proof of Theorem 1. We claim that the following inequalities hold for x^n:

$$\sum_{j=1}^{c-1} N(u^k, \mathbf{p}_j) \le N(u^k, x^n) \le \sum_{j=1}^{c-1} N(u^k, \mathbf{p}_j) + (k-1)(c-1) + |\mathbf{t}_x|. \tag{3}$$

The first inequality follows from the fact that the phrases are distinct, and they parse the head of x^n. The second inequality follows from the fact that an occurrence of u^k is either completely contained in a nontrivial phrase of the parsing, or it spans a phrase boundary, or it is contained in the tail of x^n. To span a boundary, an occurrence of u^k must start at most $k-1$ locations before the end of a nontrivial phrase. Substitute y^n for x^n in (3), and call the resulting inequalities $(3)_y$. Clearly, $(3)_y$ holds for $y^n \in \mathcal{U}_{x^n}$, since y^n has the same set of phrases as x^n, and the sequences have tails of the same length. Thus, it follows from (3) and $(3)_y$ that

$$|N(u^k, x^n) - N(u^k, y^n)| \le (k-1)(c-1) + |\mathbf{t}_x|, \quad \forall y^n \in \mathcal{U}_{x^n}. \tag{4}$$

The claim of the theorem now follows from (4) by recalling that k is fixed and $\mathbf{t}_x$ is equal to one of the phrases, by applying parts (ii) and (iii) of Lemma 1, and finally normalizing by $n-k+1$. □

[3] The term *complete* is also used sometimes as equivalent to what we have called *full*. Here, we use the term in the narrower sense of [16]. In the binary case, a complete tree is always full. When $\alpha > 2$, we allow one internal node v at level $h-1$ to have $\deg(v) < \alpha$, which is necessary to have complete trees with any number of nodes.

[4] We assume an unbounded memory LZ78 scheme where the same dictionary is used throughout the whole sequence x^n, as opposed to the "block by block" scheme described in the original LZ78 paper [6]. The distinction between the two schemes is discussed further in Section 6.

Proof of Corollary 1. Consider sequences x^n and y^n, and a kth order probability assignment $Q^{(k)}$ satisfying the conditions of the corollary. Let $\mathcal{A}^{k+1}[Q^{(k)}] = \{u^{k+1} \in \mathcal{A}^{k+1} : Q^{(k)}(u_{k+1}|u^k) \neq 0\}$. Notice that the conditions of the corollary guarantee that $\mathcal{A}^{k+1}[Q^{(k)}]$ includes all the $(k+1)$-tuples that occur in x^n or y^n, and that $Q^{(k)}(x^k)Q^{(k)}(y^k) \neq 0$. For the sequence x^n, we have

$$Q^{(k)}(x^n) = Q^{(k)}(x^k) \prod_{i=k+1}^{n} Q^{(k)}(x_i|x_{i-k}^{i-1}) \;=\; Q^{(k)}(x^k) \prod_{u^{k+1} \in \mathcal{A}^{k+1}[Q^{(k)}]} Q^{(k)}(u_{k+1}|u^k)^{N(u^{k+1},x^n)}\,, \quad (5)$$

Taking logarithms, we obtain

$$\log Q^{(k)}(x^n) = \log Q^{(k)}(x^k) + \sum_{u^{k+1} \in \mathcal{A}^{k+1}[Q^{(k)}]} N(u^{k+1},x^n) \log Q^{(k)}(u_{k+1}|u^k)\,. \quad (6)$$

An analogous expression for $Q^{(k)}(y^n)$ is obtained by substituting y for x everywhere in (6); we refer to the equation resulting from this substitution as $(6)_y$. By the discussion preceding (5), all the logarithms taken in (6) and $(6)_y$ are well defined. Subtracting $(6)_y$ from (6), we obtain

$$\log \frac{Q^{(k)}(x^n)}{Q^{(k)}(y^n)} = \log \frac{Q^{(k)}(x^k)}{Q^{(k)}(y^k)} + \sum_{u^{k+1} \in \mathcal{A}^{k+1}[Q^{(k)}]} \Big(N(u^{k+1},x^n) - N(u^{k+1},y^n)\Big) \log Q^{(k)}(u_{k+1}|u^k)\,.$$

Letting $\sigma_0 = \max_{u^k,v^k} |\log Q^{(k)}(u^k) - \log Q^{(k)}(v^k)|$, where the maximum is taken over all pairs u^k, v^k such that $Q^{(k)}(u^k)Q^{(k)}(v^k) \neq 0$, we obtain

$$\left|\log \frac{Q^{(k)}(x^n)}{Q^{(k)}(y^n)}\right| \leq \sigma_0 \;- \sum_{u^{k+1} \in \mathcal{A}^{k+1}[Q^{(k)}]} \Big|N(u^{k+1},x^n) - N(u^{k+1},y^n)\Big| \log Q^{(k)}(u_{k+1}|u^k). \quad (7)$$

It follows from the proof of Theorem 1 that $\frac{1}{n}\Big|N(u^{k+1},x^n) - N(u^{k+1},y^n)\Big|$ is uniformly upper-bounded by $\sigma_1/\log n$ for some constant σ_1 and sufficiently large n. Hence, denoting

$$\sigma_2 = \sum_{u^{k+1} \in \mathcal{A}^{k+1}[Q^{(k)}]} -\log Q^{(k)}(u^{k+1}|u_k),$$

it follows from (7) that

$$\left|\frac{1}{n}\log \frac{Q^{(k)}(x^n)}{Q^{(k)}(y^n)}\right| \leq \frac{\sigma_0}{n} + \frac{\sigma_1\sigma_2}{\log n}\,,$$

where σ_0, σ_1, and σ_2 are independent of n, x^n, and y^n. □

A kth order probability assignment of particular interest is the one defined by the kth order conditional empirical distribution of the sequence x^n itself, namely,

$$\hat{Q}_{x^n}^{(k)}(a|u^k) = \frac{N(u^k a,x^n)}{N(u^k,x^n)}\,,\; u^k \in \mathcal{A}^k,\; a \in \mathcal{A},\; N(u^k,x^n) > 0, \quad (8)$$

with initial condition $\hat{Q}^{(k)}_{x^n}(x^k) = 1$. The empirical distribution $\hat{Q}^{(k)}_{x^n}$ is the maximum likelihood (ML) estimator of a kth order probability model for x^n, i.e., it assigns to x^n the maximum probability the sequence can attain with such a model. The kth order *conditional empirical entropy* of x^n is defined as

$$\hat{H}_k(x^n) = -\log \hat{Q}^{(k)}_{x^n}(x^n)\,. \tag{9}$$

The corresponding entropy rate, in bits per α-ary symbol is

$$\hat{\mathbf{H}}_k(x^n) = \frac{1}{n}\hat{H}_k(x^n)\,. \tag{10}$$

We will prove that the kth order empirical entropy rate of sequences of the same universal type also converges. First, we need two technical lemmas.

Lemma 3 *Let P and Q be distributions over a finite alphabet B, and δ a number such that $0 \le \delta \le 1/(2|B|)$. If $\|P - Q\|_\infty \le \delta$ then $|H(P) - H(Q)| \le |B|\delta \log \delta^{-1}$.*

Proof. If $\|P - Q\|_\infty \le \delta$ then $\|P - Q\|_1 \le |B|\delta$, where $\|P - Q\|_1$ is the variational (L_1) distance bewtween P and Q. The result now follows by applying [3, Theorem 16.3.2] (cf. also [1, Lemma 2.7]). □

Lemma 4 *Let $\{x^n\}$ and $\{y^n\}$ be sequences such that $\|\hat{P}^{(k)}_{x^n} - \hat{P}^{(k)}_{y^n}\|_\infty \le \delta$ and $\|\hat{P}^{(k+1)}_{x^n} - \hat{P}^{(k+1)}_{y^n}\|_\infty \le \delta$ for some $k \ge 1$ and $0 \le \delta \le 1/(2\alpha^{k+1})$. Then,*

$$|\hat{\mathbf{H}}_k(x^n) - \hat{\mathbf{H}}_k(y^n)| \le \alpha^k(\alpha+1)\delta \log \delta^{-1} + O(\log n/n).$$

Proof. From (8), (10), and (6), we have[5]

$$\begin{aligned}
\hat{\mathbf{H}}_k(x^n) &= -\frac{1}{n}\log \hat{Q}^{(k)}_{x^n}(x^n) = -\frac{1}{n}\sum_{u^{k+1}} N(u^{k+1}, x^n) \log Q^{(k)}(u_{k+1}|u^k) \\
&= -\frac{1}{n}\sum_{u^{k+1}} N(u^{k+1}, x^n)\left(\log N(u^{k+1}, x^n) - \log N(u^k, x^n)\right) \\
&= -\frac{1}{n}\sum_{u^{k+1}} N(u^{k+1}, x^n) \log N(u^{k+1}, x^n) + \frac{1}{n}\sum_{u^{k+1}} N(u^{k+1}, x^n) \log N(u^k, x^n) \\
&= -\frac{1}{n}\sum_{u^{k+1}} N(u^{k+1}, x^n) \log N(u^{k+1}, x^n) + \frac{1}{n}\sum_{u^{k}} N(u^{k+1}, x^n) \log N(u^k, x^n) \\
&= -\frac{n-k}{n}\left(\log(n-k) - H(\hat{P}^{(k+1)}_{x^n})\right) + \frac{n-k+1}{n}\left(\log(n-k+1) - H(\hat{P}^{(k)}_{x^n})\right) \\
&= H(\hat{P}^{(k+1)}_{x^n}) - H(\hat{P}^{(k)}_{x^n}) + O(\log n/n).
\end{aligned} \tag{11}$$

[5]Equation (11) can be seen as an empirical version of the chain rule for random variables, namely, $H(Y|X)$=$H(X,Y)$−$H(X)$. A similar empirical expression is given in [17], which does not include the $O(\log n/n)$ term, and is actually derived from the chain rule by making a cyclic (shift-invariance) assumption on the conditional probabilities. Here, since our kth order assignments admit arbitrary probabilities for the initial state, we derive the empirical rule from first principles, and do not rely on the stochastic rule.

A similar expression holds for y^n, and we can write

$$\begin{aligned} |\hat{\mathbf{H}}_k(x^n) - \hat{\mathbf{H}}_k(y^n)| &= |H(\hat{P}_{x^n}^{(k+1)}) - H(\hat{P}_{x^n}^{(k)}) - H(\hat{P}_{y^n}^{(k+1)}) + H(\hat{P}_{y^n}^{(k)}) + O(\log n/n)| \\ &\leq |H(\hat{P}_{x^n}^{(k+1)}) - H(\hat{P}_{y^n}^{(k+1)})| + |H(\hat{P}_{x^n}^{(k)}) - H(\hat{P}_{y^n}^{(k)})| + O(\log n/n) \\ &\leq \alpha^{k+1}\delta \log \delta^{-1} + \alpha^k \delta \log \delta^{-1} + O(\log n/n)\,, \end{aligned} \tag{12}$$

where the last inequality follows from the assumptions of the lemma, and from Lemma 3, recalling that $\hat{P}_{x^n}^{(k+1)}$ and $\hat{P}_{y^n}^{(k+1)}$ are distributions on $\mathcal{A}^{k+1}$, while $\hat{P}_{x^n}^{(k)}$ and $\hat{P}_{y^n}^{(k)}$ are defined on $\mathcal{A}^k$. The claim now follows by collecting terms on the right hand side of (12). □

Corollary 2 *Let $\{x^n\}$ and $\{y^n\}$ be sequences such that $y^n \in \mathcal{U}_{x^n}$. Then, for any $k \geq 0$, we have*

$$\lim_{n\to\infty} \left| \hat{\mathbf{H}}_k(x^n) - \hat{\mathbf{H}}_k(y^n) \right| = 0,$$

with convergence rate $O(\log\log / \log n)$, uniformly in x^n and y^n.

Proof. The claim follows from Theorem 1 and Lemma 4, with convergence rate determined by the convergence rate in Theorem 1, to which the transformation $\delta \to \delta \log \delta^{-1}$ is applied. □

Sequences of the same type have, by definition of the UTC, exactly the same LZ78 code length. Corollary 2 says that their kth order empirical entropy rates will also converge, for all k. The two measures of compressibility, however, do not always coincide, as we shall see in an explicit example in Section 6.

3 The size of the universal type class

In this section, we characterize the number of sequences in a universal type class. First, we present, in Section 3.1, a recursion and exact combinatorial characterization of the UTC size. The question is closely related to classical problems in computer science, namely, labelings of a tree that preserve the tree-induced partial order, and the notion of a *heap*, a fundamental data structure for searching and sorting [18]. Although some of the results presented could be derived from analogous results in these areas, we give self-contained proofs for completeness, and to cast the results in their most generality for the setting of universal types. These results will be used to derive the information-theoretic properties and algorithms presented in subsequent sections. In Section 3.2, we study the asymptotic behavior of the UTC size as a function of global parameters of the parsing of x^n, and relate it to the compressibility of the sequence.

3.1 Combinatorial characterization

Each sequence in the UTC of x^n is determined by some permutation of the order of the phrases in T_{x^n}. Not all the possible permutations are allowed, though, since, by the rules defining the

parsing, a phrase must always precede its extensions. Thus, only permutations that respect the prefix partial order are valid.

Recall that if $a \in \mathcal{A} \cap T_{x^n}$, then $T^{(a)}$ denotes the subtree of T_{x^n} rooted at the child a of the root (if $a \in \mathcal{A} \setminus T_{x^n}$, then $T^{(a)}$ is empty). We refer to $T^{(a)}$ as a *main subtree* of T_{x^n}. If $T^{(a)}$ is not empty, let $x^n[a]$ denote the sub-sequence of x^n constructed by concatenating all the phrases of x^n that start with a (preserving their order in x^n), after eliminating the initial symbol a from each phrase. Clearly, we have $\mathcal{U}(T^{(a)}) = \mathcal{U}_{x^n[a]}$. Denote the number of nodes in $T^{(a)}$ by c_a, $a \in \mathcal{A}$. It will be convenient to extend our notations also to empty subtrees; when $T^{(a)}$ is empty, there are no phrases starting with a, we have $c_a = 0$, and we adopt the convention that $|\mathcal{U}(T^{(a)})| = 1$. Clearly, we have $c(x^n) = c_0 + c_1 + \ldots + c_{\alpha-1} + 1$.

A valid permutation of phrases defining a sequence $y^n \in \mathcal{U}_{x^n}$ must result from valid permutations of the phrases in each of the subtrees $T^{(a)}$, $a \in \mathcal{A}$. The resulting ordered sublists of phrases can be freely interleaved to form a valid ordered list of phrases for y^n, since there is no order constraint between phrases in different subtrees. The number of possible interleavings is given by the multinomial coefficient

$$M(c_0, c_1, \ldots, c_{\alpha-1}) = \frac{(c_0 + c_1 + \cdots + c_{\alpha-1})!}{c_0! c_1! \ldots c_{\alpha-1}!}\,, \tag{13}$$

namely, the number of ways to merge α ordered lists of respective sizes $c_0, c_1, \ldots, c_\alpha$ into one list of size $c-1 = \sum_{a\in\mathcal{A}} c_i$, while preserving the order of each respective sublist. By extension of our sequence notation, we denote by $c_0^{\alpha-1}$ the vector of integers $(c_0, c_1, \ldots, c_{\alpha-1})$, and we use $M(c_0^{\alpha-1})$ as shorthand for $M(c_0, c_1, \ldots, c_{\alpha-1})$.

Given the sublists and their interleaving, to completely specify y^n, we must also define its tail, which can be any phrase of length $|\mathbf{t}_x|$ (at least one such phrase exists, namely, $\mathbf{t}_x$ itself). Let $\tau(x^n)$ denote the number of nodes at level $|\mathbf{t}_x|$ in T_{x^n}. The foregoing discussion is summarized in the following theorem, which presents a recursion for the size of a UTC.

Proposition 1 *Let x^n be a sequence, and let $T^{(a)}$, $a \in \mathcal{A}$, denote the main subtrees of T_{x^n}. Then,*

$$|\mathcal{U}_{x^n}| = \left(\prod_{a\in\mathcal{A}} \left|\mathcal{U}(T^{(a)})\right| \right) M(c_0^{\alpha-1})\, \tau(x^n). \tag{14}$$

Notice that when (14) is used recursively, all recursion levels except the outermost one deal with natural UTCs. Therefore, a nontrivial factor $\tau(x^n)$ occurs only at the outermost application of (14). Moreover, we have $\tau(x^n) \le c$, which will turn out to be a negligible factor in the asymptotics of interest of $|\mathcal{U}_{x^n}|$. Therefore, most of our discussions will assume that $\mathbf{t}_x = \lambda$, and will focus on natural types. The asymptotic results, however, will hold for arbitrary strings and their universal types. We will point that fact out and justify it when these results are proved.

Variants of the recursion (14), especially for the case of complete binary trees, have been extensively studied in connection to heaps. See, e.g., [18, 19, 20] and references therein.

Let $c_{\mathbf{p}}$, $\mathbf{p} \in T_{x^n}$, denote the number of phrases in T_{x^n} that have the phrase $\mathbf{p}$ as a prefix, or equivalently, the number of nodes in the subtree of T_{x^n} rooted at $\mathbf{p}$. This definition generalizes the

previous notation c_a, where a symbol $a \in \mathcal{A}$ is regarded as a string of length one. For a tree T, define

$$D(T) = \prod_{\mathbf{p} \in T} c_{\mathbf{p}}. \tag{15}$$

The following proposition presents a more explicit expression for $|\mathcal{U}_{x^n}|$.

Proposition 2 *Let x^n be a sequence with $\mathbf{t}_x = \lambda$. Then,*

$$|\mathcal{U}_{x^n}| = \frac{c\,!}{D(T)}\,. \tag{16}$$

Proof. We prove the claim of the corollary by induction on c. For $c = 1$, we have $x^n = \lambda$, and $|\mathcal{U}_\lambda| = 1$, which trivially satisfies the claim. Assume the claim is satisfied by all strings with c' phrases for $1 \le c' < c$, and let T be the parsing tree of a sequence x^n with c phrases. Let $\mathcal{A}' \subseteq \mathcal{A}$ be the set of symbols a such that $c_a > 0$. We have $0 < c_a < c = 1 + \sum_{a \in \mathcal{A}'} c_a$, and, therefore, the UTCs corresponding to $T^{(a)}$, $a \in \mathcal{A}'$, satisfy the induction hypothesis. Substituting the value given by the induction hypothesis for $|\mathcal{U}(T^{(a)})|$ in (14), recalling also that $|\mathcal{U}(T^{(a)})| = 1$ when $a \in \mathcal{A} \setminus \mathcal{A}'$ and that $\tau(x^n) = 1$, and recalling the definition of $M(c_0^{\alpha-1})$ from (13), we obtain

$$|\mathcal{U}_{x^n}| = \left(\prod_{a \in \mathcal{A}'} \frac{c_a!}{\prod_{\mathbf{p} \in T^{(a)}} c_{\mathbf{p}}} \right) \left(\frac{(c-1)!}{\prod_{a \in \mathcal{A}'} c_a!} \right) = \frac{(c-1)!}{\prod_{a \in \mathcal{A}'} \prod_{\mathbf{p} \in T^{(a)}} c_{\mathbf{p}}} = \frac{(c-1)!}{\prod_{\mathbf{p} \in T_{x^n} \setminus \{\lambda\}} c_{\mathbf{p}}} = \frac{c!}{\prod_{\mathbf{p} \in T_{x^n}} c_{\mathbf{p}}}\,.$$

□

The expression at the right hand side of (16) is known [18, p. 67] as the number of ways to label the nodes of a tree of size c with the numbers $0, 1, \ldots, c-1$, so that the label of a node is less than that of any descendant. In our context, letting the label of a node be the ordinal number of the corresponding phrase in a permutation of the phrases of x^n, valid tree labelings correspond to valid phrase permutations, and the result in [18, p. 67] is equivalent to Proposition 2.

Example 2. For the tree T in Figure 1, we have $c = c_\lambda = 6$, c_0=2, c_1=3, $c_{00} = c_{10} = c_{11} = 1$. Therefore,

$$|\mathcal{U}(T)| = \frac{6!}{2 \cdot 3 \cdot 6} = 20.$$

□

Example 3. Consider the tree shown in Figure 2, which contains a chain of nodes of degree one starting at the root and extending to depth $k \ge 1$, where a subtree T' is rooted. It follows readily from Proposition 2 that, in this case, we have $|\mathcal{U}(T)| = |\mathcal{U}(T')|$, independently of k. In particular, when T' is a trivial tree consisting of just a root node, we have $|\mathcal{U}(T)| = 1$ (this is the "linear" tree discussed in the remarks following Lemma 1). □

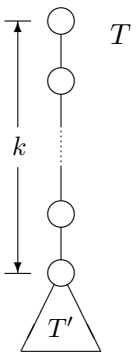

Figure 2: Chain of nodes from the root.

Propositions 1 and 2 provide exact expressions for the size of the UTC of a given sequence x^n, based on the detailed structure of its parsing tree. These expressions, however, do not provide direct insight into the asymptotic behavior of the UTC size as the sequence length increases. We next derive an asymptotic characterization of the UTC size as a function of coarser global parameters of the sequence and its parsing tree.

3.2 Asymptotic behavior of the UTC size

First, we present some auxiliary definitions and lemmas that will aid in the derivation.

Let I and L denote, respectively, the set of internal nodes and leaves of a tree T. The two subsets can also be characterized as $I = \{\mathbf{p} \mid \mathbf{p} \in T,\ c_{\mathbf{p}} > 1\}$ and $L = \{\mathbf{p} \mid \mathbf{p} \in T,\ c_{\mathbf{p}} = 1\}$.

Lemma 5 *We have*

$$\sum_{\mathbf{p}\in T} c_{\mathbf{p}} = n + c, \tag{17}$$

and

$$\sum_{\mathbf{p}\in I} c_{\mathbf{p}} = n + c - \ell. \tag{18}$$

Proof. Each node $\mathbf{p}$ contributes a unit to the sum in (17) for each subtree it belongs to, or equivalently, for each of its ancestors in T (including itself). Therefore, the contribution of $\mathbf{p}$ to the sum is $|\mathbf{p}| + 1$, and we have

$$\sum_{\mathbf{p}\in T} c_{\mathbf{p}} = \sum_{\mathbf{p}\in T} (|\mathbf{p}| + 1) = \sum_{\mathbf{p}\in T} |\mathbf{p}| + c = n + c.$$

As to the sum in (18), we have

$$\sum_{\mathbf{p}\in I} c_{\mathbf{p}} = \sum_{\mathbf{p}\in T} c_{\mathbf{p}} - \sum_{\mathbf{p}\in L} c_{\mathbf{p}} = n + c - \ell.$$

□

It is well known (see, e.g., [16, p. 595]) that a tree T with ℓ leaves and c nodes is full if and only if

$$(\alpha - 1)c = \alpha\ell - 1. \tag{19}$$

In a tree that is not full, some internal nodes are "missing" outgoing branches. The total number of such missing branches is $d = (\alpha - 1)c - \alpha\ell + 1$, which is always nonnegative. In particular, we have

$$c > c - \ell = \frac{c - 1 + d}{\alpha} \geq \frac{c - 1}{\alpha}\ . \tag{20}$$

Lemma 6 *Let T be a tree with $c \geq 2$. Then,*

$$\log D(T) \leq (c - \ell)\left(\frac{\log n}{\log c} - 1\right)\log c + \gamma c, \tag{21}$$

for some constant $\gamma > 0$.

Proof. Since $c_{\mathbf{p}} = 1$ whenever $\mathbf{p} \in T \setminus I$, we have

$$D(T) = \prod_{\mathbf{p}\in T} c_{\mathbf{p}} = \prod_{\mathbf{p}\in I} c_{\mathbf{p}}.$$

Thus, $D(T)$ is a product of $c-\ell$ positive real numbers whose sum is constrained by (18). Such a product is maximized when all the factors are equal. Hence, $D(T)$ can be upper-bounded as follows:

$$D(T) \leq \left(\frac{\sum_{\mathbf{p}\in I} c_{\mathbf{p}}}{c - \ell}\right)^{c-\ell} = \left(\frac{n + c - \ell}{c - \ell}\right)^{c-\ell}.$$

Taking logarithms, recalling that $c \leq n$, applying (20), and performing some elementary algebraic manipulations, we obtain

$$\begin{aligned}\log D(T) &\leq (c - \ell)\Big(\log(n + c - \ell) - \log(c - \ell)\Big) < (c - \ell)\left(\log(2n) - \log\Big(\frac{c - 1}{\alpha}\Big)\right)\\ &\leq (c - \ell)\left(\frac{\log n}{\log c} - \frac{\log(c - 1)}{\log c}\right)\log c + \gamma_1 c,\end{aligned} \tag{22}$$

for some constant $\gamma_1 > 0$. The claim of the lemma now follows by writing $\log(c-1)/\log c > 1 - 2/(c \ln c)$, which holds for $c \geq 2$, after a few additional elementary manipulations. □

Lemma 7 *Let T be a tree of height h and path length n. Then,*

$$n \geq \frac{h(h + 1)}{2}\ , \tag{23}$$

and

$$D(T) \geq (h + 1)!\,. \tag{24}$$

Proof. The bound on n follows from the fact that T has at least one node at each depth j, $0 \le j \le h$. The bound on $D(T)$ follows similarly, by observing that a node $\mathbf{p}$ at depth j on the longest path of the tree is the root of a subtree of size $c_{\mathbf{p}} \ge h - j + 1$, $0 \le j \le h$. Hence, $D(T) = \prod_{\mathbf{p} \in T} c_{\mathbf{p}} \ge \prod_{j=0}^{h} (h - j + 1) = (h+1)!$. □

In the sequel, all expressions involving limits implicitly assume $n \to \infty$ (and, thus, also $c \to \infty$ and $h \to \infty$).

Lemma 8 *Let c, ℓ, and h be the parsing parameters associated with x^n in a sequence $\{x^n\}$. Then,*

$$1 \le \underline{\lim}_n \frac{\log n}{\log c} \le \overline{\lim}_n \frac{\log n}{\log c} \le 2, \tag{25}$$

$$\overline{\lim}_n \frac{h}{c} > 0 \quad \textit{only if} \quad \overline{\lim}_n \frac{\log n}{\log c} = 2, \tag{26}$$

and

$$0 \le \underline{\lim}_n \frac{\ell}{c} \le \overline{\lim}_n \frac{\ell}{c} \le \frac{\alpha - 1}{\alpha}. \tag{27}$$

Proof. The claims follow immediately from Lemma 1(i), (23), and (20), respectively. □

Define

$$\mathcal{L}_{\mathcal{U}}(x^n) = \log |\mathcal{U}_{x^n}|, \tag{28}$$

and the corresponding normalized quantity

$$\overline{\mathcal{L}}_{\mathcal{U}}(x^n) = n^{-1} \mathcal{L}_{\mathcal{U}}(x^n). \tag{29}$$

We will also write $\mathcal{L}_{\mathcal{U}}(T)$ and $\overline{\mathcal{L}}_{\mathcal{U}}(T)$, where T is a tree, when referring to natural types. The following theorem gives our main asymptotic result on the size of a universal type class.

Theorem 2 *Let $\{x^n\}$ be an arbitrary sequence with $n \ge 1$. Then,*

$$(1 - \beta - o(1))\, c \log c \le \mathcal{L}_{\mathcal{U}}(x^n) \le (1 - \eta - o(1))\, c \log c, \tag{30}$$

where

$$\beta = \Big(1 - \frac{\ell}{c}\Big)\Big(\frac{\log n}{\log c} - 1\Big), \tag{31}$$

and

$$\eta = \frac{(h+1)\log(h+1)}{c \log c}. \tag{32}$$

Moreover, we have $0 < \beta, \eta \le 1$,

$$\overline{\lim}_n \beta > 0 \quad \textit{if and only if} \quad \overline{\lim}_n \log n / \log c > 1,$$

and

$$\overline{\lim}_n \eta > 0 \quad \textit{only if} \quad \lim_n \log n / \log c = 2.$$

Thus, $\lim_n \beta = \lim_n \eta = 0$ whenever $\lim_n \log n / \log c = 1$.

Proof. Assume first that $\mathbf{t}_x = \lambda$. To prove the upper bound, we use Proposition 2, and (24) in Lemma 7, writing

$$\begin{aligned}\mathcal{L}_{\mathcal{U}}(x^n) &= \log \frac{c!}{D(T_{x^n})} \leq \log c! - \log(h+1)! \\ &\leq c\log c - c\log e + O(\log c) - (h+1)\log(h+1) \\ &= \Big(1 - \frac{(h+1)\log(h+1)}{c\log c} - o(1)\Big)\, c\log c = \Big(1 - \eta - o(1)\Big)\, c\log c.\end{aligned}$$

It is verified by direct inspection that $0 < \beta, \eta \leq 1$, since we have $\ell > 0$, $\log n/\log c < 2$, and $h < c$. By the definition of η in (32), we can have $\overline{\lim}_n \eta > 0$ only if $\overline{\lim}_n h/c > 0$, and, by (26) in Lemma 8, this is possible only if $\overline{\lim}_n \log n/\log c = 2$.

To prove the lower bound, we take logarithms on both sides of (16), and apply the result of Lemma 6, and Stirling's approximation, as follows.

$$\begin{aligned}\mathcal{L}_{\mathcal{U}}(x^n) &= \log c! - \log D(T_{x^n}) \geq c\log c - c\log e - (c-\ell)\Big(\frac{\log n}{\log c} - 1\Big)\log c - \gamma_1 c \\ &= \Big(1 - \Big(1 - \frac{\ell}{c}\Big)\Big(\frac{\log n}{\log c} - 1\Big)\Big)\, c\log c - \gamma_2 c = \Big(1 - \beta - o(1)\Big)\, c\log c,\end{aligned}$$

where γ_1 and γ_2 are appropriate positive constants. The asymptotic behavior of β is controlled by the two factors at the right hand side of (31). Since $1 - \ell/c$ is bounded, it follows from (25) that β vanishes in the limit unless $\overline{\lim}_n \log n/\log c > 1$. Conversely, by (27), the ratio ℓ/c cannot exceed $(\alpha-1)/\alpha$ in the limit. Thus, if $\overline{\lim}_n \log n/\log c = r$, with $1 < r \leq 2$, then $\overline{\lim}_n \beta \geq (r-1)/\alpha > 0$.

When $\mathbf{t}_x \neq \lambda$, we recall, from Proposition 1 and the discussion that follows it, that $|\mathcal{U}_{x^n}| = |\mathcal{U}(T_{x^n})|\,\tau(x^n) \leq |\mathcal{U}_{T_{x^n}}|\,c$. Therefore, we have

$$\mathcal{L}_{\mathcal{U}}(T_{x^n}) \leq \mathcal{L}_{\mathcal{U}}(x^n) \leq \mathcal{L}_{\mathcal{U}}(T_{x^n}) + \log c.$$

Upon normalization by $c\log c$, the term $\log c$ is absorbed into the $o(1)$ terms in (30). Hence, the bounds of the theorem hold for all sequences x^n. □

Example 4. Consider a complete, perfectly balanced α-ary tree, T of height m, i.e., a tree with α^j nodes at each depth j, $0 \leq j \leq m$. The size of the universal type class can be determined quite precisely in this case. Writing $c_m = |T| = (\alpha^{m+1} - 1)/(\alpha - 1)$, and $d_m = \log D(T)$, it is readily verified that d_m satisfies the recursion

$$d_m = \alpha\, d_{m-1} + \log c_m, \quad m \geq 1, \quad d_0 = 0.$$

The recursion is solved using standard difference equation methods, yielding

$$d_m = f\, c_m - g\, m + o(m),$$

where $f = (\log\alpha)/(\alpha-1) + (\log\alpha - \log(\alpha-1))/\alpha$, and $g = (\log\alpha)/(\alpha-1)$. Combining with (16), we obtain

$$\mathcal{L}_{\mathcal{U}}(T) = c\log c - (f+1)c + (\frac{1}{2} + \frac{1}{\alpha-1})\log c + o(\log c).$$

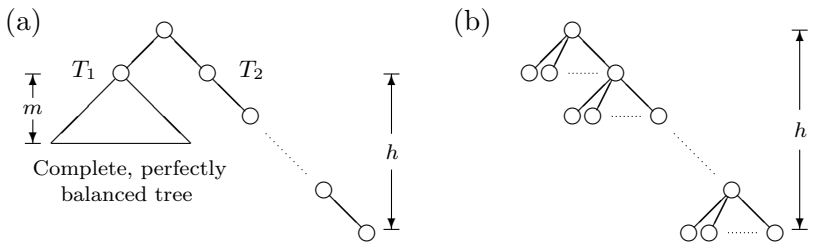

Figure 3: Two α-ary trees

The tree T in this example corresponds to the parsing of an α-ary *counting sequence*, defined as the concatenation of all α-ary sequences of each length i, $1 \leq i \leq m$, in increasing length order (these sequences are related to, although not identical, to the sequence of digits in a α-ary *Champerknowne number* [21], [22, A033307]). Counting sequences are known to be "hardest" to compress by the LZ78 algorithm [6]. The bounds in Theorem 2 are asymptotically tight in this case, as it is readily verified that $\beta \to 0$ and $\eta \to 0$ as $m \to \infty$. □

Example 5. Consider the tree T shown in Figure 3(a), where the subtree T_1 is a complete perfectly balanced tree of height m, and the subtree T_2 is a "linear" tree of height h. Let $c_1 = |T_1|$, and $c_2 = |T_2| = h + 1$. By the analysis in Example 4, we have $\mathcal{L}_{\mathcal{U}}(T_1) = c_1 \log c_1 + O(c_1)$, and, by Example 3, we have $\mathcal{L}_{\mathcal{U}}(T_2) = 0$. Using the recursion (14), we obtain

$$\mathcal{L}_{\mathcal{U}}(T) = c_1 \log c_1 + \log \left(\begin{array}{c} c_1 + h + 1 \\ h + 1 \end{array} \right) + O(c_1) \, . \tag{33}$$

Assume now that $h = \mu c_1$, for some $\mu > 0$. Then, we have $c = (1 + \mu)c_1 + 2$, and the binomial coefficient in (33) can be approximated by $\exp\left(\xi c + o(c)\right)$ for some $\xi > 0$ (see, e.g., [23, Ch. 10]). Hence, (33) can be rewritten as

$$\mathcal{L}_{\mathcal{U}}(T) = (1 + \mu)^{-1} c \log c + O(c).$$

On the other hand, it is readily verified that $\log n / \log c \to 2$, and $\ell / c \to (1 + \mu)^{-1}$ in this example, and, hence, $(1 - \beta) \to (1 + \mu)^{-1}$. Similarly, since $m = O(\log c_1) \ll h$ for large values of c_1, the height of T is $h + 2$, and we have $1 - \eta \approx 1 - h/c \to (1 + \mu)^{-1}$. We conclude that the bounds in Theorem 2 are asymptotically tight in this case. Observe also that the example leads to the construction of sequences x^n such that $\mathcal{L}_{\mathcal{U}}(x^n) = \gamma \, c \log c + O(c)$ for all $\gamma \in [0, 1]$ ($\gamma = 0$ is achieved by setting $m = o(\log h)$, which corresponds to $\mu \to \infty$). □

Example 6. Consider the α-ary tree of height h shown in Figure 3(b), which has one internal node and $\alpha - 1$ leaves at each level j, $1 \leq j \leq h$. For this tree, we have $n = (c^2 + (\alpha - 2)c - \alpha + 1)/2\alpha$, and $\ell = ((\alpha - 1)c + 1)/\alpha$. Thus, $\beta \to \alpha^{-1}$. Also, $c = \alpha h + 1$, and, therefore, $\eta \to \alpha^{-1}$. The bounds of Theorem 2 are asymptotically tight, yielding $\mathcal{L}_{\mathcal{U}}(T) = (1 - \alpha^{-1}) c \log c + O(c)$. This can also be verified by direct computation of $|\mathcal{U}(T)|$ from (16). □

Corollary 3 *Let $\{x^n\}$ be an arbitrary sequence such that $\underline{\lim}_n \overline{\mathcal{L}}_{LZ}(x^n) > 0$. Then,*

$$\mathcal{L}_{\mathcal{U}}(x^n) = \Big(1 + o(1)\Big)\, c \log c\,.$$

Proof. If $\underline{\lim}_n \overline{\mathcal{L}}_{\mathrm{LZ}}(x^n) > 0$ then $n = O(c \log c)$, and, thus $\lim_n \log n / \log c = 1$. By Theorem 2, this implies that $\lim_n \beta = \lim_n \eta = 0$, and, therefore, $\mathcal{L}_{\mathcal{U}}(x^n) = (1 + o(1))c \log c$. □

Corollary 4 *Let $\{x^n\}$ be an arbitrary sequence. Then,*

$$\lim_n \Big|\overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \overline{\mathcal{L}}_{LZ}(x^n)\Big| = 0\,.$$

Proof. Assume the claim is not true. Then, there exists an infinite sequence $\{x^n\}$, for a possibly sparse sequence of values of n, and a constant $\varepsilon > 0$, such that $|\overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \overline{\mathcal{L}}_{\mathrm{LZ}}(x^n)| > \varepsilon$ for all n in the sequence. But, by Lemma 2 and the upper bound in Theorem 2, this implies that $\overline{\mathcal{L}}_{\mathrm{LZ}}(x^n) > \varepsilon'$ for some $\varepsilon' > 0$, and then by Corollary 3, we would have $\overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \overline{\mathcal{L}}_{\mathrm{LZ}}(x^n) = o(1)\overline{\mathcal{L}}_{\mathrm{LZ}}(x^n)$, contradicting the assumption, since $\overline{\mathcal{L}}_{\mathrm{LZ}}(x^n)$ is bounded. Thus, the claim of the corollary must hold. □

Corollary 4 highlights another parallel between universal types and conventional types: the size of a conventional type satisfies $\log |\mathcal{T}^{\mathcal{P}}_{x^n}| = n\hat{\mathbf{H}}(x^n)(1 + o(1))$ (cf. [24, 25, 3, 2, 26]), where $\hat{\mathbf{H}}(x^n)$ denotes the empirical entropy rate of x^n with respect to the model class $\mathcal{P}$.[6] Corollary 4 states that a similar statement is true for UTs, with the normalized LZ78 code length $\overline{\mathcal{L}}_{\mathrm{LZ}}$ playing the role of the empirical entropy rate.

Enumerative coding. An α-ary tree with c nodes can be encoded in $O(c)$ bits using, for example, variants of the *natural codes* [27] (this is one possible choice of description of the type class; the number of classes is discussed in detail in Section 4). Thus, it is possible to define an *enumerative code* [28] that will encode x^n in $\mathcal{L}'_{\mathcal{U}}(x^n) = \mathcal{L}_{\mathcal{U}}(x^n) + O(c)$ bits, by encoding the parsing tree followed by the index of x^n in an enumeration of the universal type class. Algorithms for mapping sequences to and from their indices in an enumeration of their UTC are presented in Section 5.1. For sequences $\{x^n\}$ with non-vanishing compressibility $\overline{\mathcal{L}}_{\mathrm{LZ}}$ we have, by Corollary 3, $\mathcal{L}'_{\mathcal{U}}(x^n)/\mathcal{L}_{\mathrm{LZ}}(x^n) \to 1$ as $n \to \infty$. However, for highly compressible sequences, the enumerative scheme based on UTs shows finer discrimination, and families of sequences with $\mathcal{L}_{\mathcal{U}}(x^n) = \gamma c \log c + O(c)$ for any γ in the range $0 < \gamma < 1$ can be constructed (see Example 5). For these families, we have $\mathcal{L}'_{\mathcal{U}}(x^n)/\mathcal{L}_{\mathrm{LZ}}(x^n) \to \gamma < 1$, since the LZ78 scheme always compresses to code length $\mathcal{L}_{\mathrm{LZ}}(x^n) = c \log c + O(c)$.

4 The number of universal types

One of the basic questions in studying the partition of the space $\mathcal{A}^n$ into type classes is how many classes there are. In particular, many applications of the method of types hinge on whether the number of type classes is sub-exponential in n, which is the desirable situation [2, Sec. VII].

[6] Defined, as previously done in (10) for kth order Markov models, as $\hat{\mathbf{H}}(x^n) = -n^{-1} \log P_{\hat{\boldsymbol{\Theta}}}(x^n)$, where $\hat{\boldsymbol{\Theta}}$ is the ML estimator of the parameter vector for x^n.

Let $\mathcal{N}_n$ denote the number of universal type classes for sequences of length n, and let $\widetilde{\mathcal{N}}_n$ denote the corresponding number of natural type classes.

Since a natural type is completely characterized by an α-ary tree of path length equal to n, $\widetilde{\mathcal{N}}_n$ is equal to the number of such trees. However, precise estimates for this number were not available, despite the fundamental nature of the combinatorial question. A tight asymptotic estimate was recently obtained in [15], and is being published separately, given the intrinsic interest in the combinatorial problem. The main result of [15] is the following theorem.

Theorem 3 ([15]) *The number of trees with path length equal to n is*

$$\widetilde{\mathcal{N}}_n = \exp_\alpha\Big(\frac{\alpha\,\mathbf{h}(\alpha^{-1})n}{\log n}\,\Big(1+o(1)\Big)\Big),$$

where $\mathbf{h}(u) = -u\log u - (1-u)\log(1-u)$ is the binary entropy function.

The upper bound in [15] is based on a simple coding argument for trees of path length n. The lower bound, on the other hand, is based on a combinatorial construction of large families of trees of the same path length. The number of binary trees of a given path length has also been recently studied in [29], motivated by preliminary versions of this work and of the results in [15]. The results of [29], which are based on the WKB heuristic, are consistent with the case of $\alpha = 2$ in Theorem 3.

As mentioned in Section 2, a tree of path length equal to $\tilde{n}$ can represent the parsing dictionary of sequences of length n in the range $\tilde{n} \le n \le \tilde{n} + h$, with $h \le \sqrt{2\tilde{n}}$. Conversely, for a given sequence x^n, the path length, $\tilde{n}$, of T_{x^n} is in the range $n - \sqrt{2n} \le \tilde{n} \le n$. Therefore,

$$\mathcal{N}_n \le \sum_{\tilde{n}=\lceil n-\sqrt{2n}\rceil}^{n} \widetilde{\mathcal{N}}_{\tilde{n}} \;\le\; (\sqrt{2n})\,\widetilde{\mathcal{N}}_n,$$

and, thus, by Theorem 3,

$$\log_\alpha \mathcal{N}_n \le \log_\alpha(2n) + \frac{\alpha\,\mathbf{h}(\alpha^{-1})n}{\log n}\,\big(1+o(1)\big)\ .$$

The term $\log_\alpha(2n)$ can clearly be absorbed into the $o(1)$ term. This leads to the following main result of this section.

Corollary 5 *The number of universal types for sequences of length n is*

$$\mathcal{N}_n = \exp_\alpha\Big(\frac{\alpha\,\mathbf{h}(\alpha^{-1})n}{\log n}\,\Big(1+o(1)\Big)\Big).$$

It follows from Corollary 5 that $\mathcal{N}_n$ is, indeed, sub-exponential in n.

In the classical theory of types, the number of classes is related to the redundancy of coding schemes (or, equivalently, probability assignments) that are universal in certain parametric model classes. Concretely, let $N_n^{(K)}$ denote the number of type classes for the family $\mathcal{P}_F$ of α-ary finite-state

(FS) probability assignments relative to a finite-state machine (FSM), F, with K states [30].[7] It is known ([30], attributed to N. Alon) that, under some mild regularity conditions on F, $\log N_n^{(K)} = (\alpha-1)K\log n + O(1)$. This matches the redundancy of probability assignments that are pointwise universal in $\mathcal{P}_F$, which is, up to lower order terms, $\frac{1}{2}(\alpha-1)K\log n \approx \frac{1}{2}\log N_n^{(K)}$. For universal type classes, $\log \mathcal{N}_n$ as given in Corollary 5 matches, up to constant multiplication, the redundancy of the LZ78 scheme for finite-memory sources, which is $O(n/\log n)$ (unnormalized) [31, 32].

5 Selecting sequences from a universal type class

The recursion (14) is helpful for deriving efficient procedures for enumerating $\mathcal{U}_{x^n}$, and for drawing random sequences from the class with uniform probability. Enumeration of the type class is discussed first. It allows for implementation of the enumerative coding scheme discussed in Section 3.2, and also provides one method for drawing a random sequence with (almost) uniform probability. An alternative method for random selection, which can attain exactly uniform probability, is described in Section 5.2. Either method can be used to implement the universal simulation scheme for individual sequences discussed in Section 6.

5.1 Enumeration of sequences in a universal type class

A one-to-one function $J_{x^n} : \mathcal{U}_{x^n} \to \{0, 1, \ldots, |\mathcal{U}_{x^n}| - 1\}$ is called an *enumeration* of $\mathcal{U}_{x^n}$. We are interested in defining an enumeration J_{x^n}, together with efficient algorithms for evaluating $J_{x^n}(y^n)$ for any $y^n \in \mathcal{U}_{x^n}$, and the inverse function $J_{x^n}^{-1}$ that reconstructs a sequence from its index.

The conventional α-ary type class $\mathcal{T}[c_0^{\alpha-1}]$ is defined as the set of vectors of length $c_0 + c_1 + \cdots + c_{\alpha-1}$ with c_i occurrences of the symbol i, $0 \le i < \alpha$. In defining J_{x^n} and $J_{x^n}^{-1}$, we will make use, as primitives, of functions that enumerate $\mathcal{T}[c_0^{\alpha-1}]$, namely, one-to-one functions

$$F[c_0^{\alpha-1}] \,:\, \mathcal{T}[c_0^{\alpha-1}] \,\to\, \{0, 1, \ldots, M(c_0^{\alpha-1}) - 1\}$$

and

$$F[c_0^{\alpha-1}]^{-1} \,:\, \{0, 1, \ldots, M(c_0^{\alpha-1}) - 1\} \,\to\, \mathcal{T}[c_0^{\alpha-1}]\,.$$

Such functions are readily constructed by iterating well known methods for enumerating and ranking combinations (see, e.g., [16, 33, 28]), based mostly on the *combinatorial number system* [16], and they are of moderate (polynomial) complexity.

For a sequence $x^n \in \mathcal{A}^n$, let $w(x^n) \in \mathcal{A}^{c-1}$ denote the sub-sequence built from the first symbols of the non-null phrases of x^n, in the same order as the phrases they come from. We recall that $\tau(x^n)$ denotes the number of phrases in T_{x^n} that are of the same length as $\mathbf{t}_x$. Let $\psi : \{\, \mathbf{p} \in T_{x^n} \,:\, |\mathbf{p}| = |\mathbf{t}_x|\} \to \{0, 1, \ldots, \tau(x^n)-1\}$ denote an enumeration of possible tails of sequences in $\mathcal{U}_{x^n}$.

Algorithm E in Figure 4 implements an enumeration J_{x^n} of $\mathcal{U}_{x^n}$. The workings of the algorithm are straightforward: given an input sequence y^n, the indices of the sub-sequences $y^n[a]$ in their

[7] This is a fairly general setting: $\mathcal{P}_F$ could consist, for example, of all kth order finite-memory (Markov) probability assignments, with $K = \alpha^k$.

ALGORITHM E.
Input: Sequence y^n, parsing tree $T = T_{x^n}$.
Output: $J_{x^n}(y^n)$, index of y^n in $\mathcal{U}_{x^n}$.

1. `If` $n = 0$, `return` $J_{x^n}(y^n) = 0$.
2. `Let` $u_a = |\mathcal{U}_{x^n[a]}|$, $a \in \mathcal{A}$.
3. `For each` $a \in \mathcal{A}$,
   `let` $j_a = J_{x^n[a]}(y^n[a])$. `// Recursive call`
4. `Let` $j = j_0 + u_0 j_1 + u_0 u_1 j_2 + \ldots + u_0 u_1 \ldots u_{\alpha-2} j_{\alpha-1}$.
5. `Let` $f = F[c_0^{\alpha-1}](w(y^n))$.
6. `Let` $t = \psi(\mathbf{t}_y)$.
7. `Return` $J_{x^n}(y^n) = t + \tau(x) f + \tau(x) M(c_0^{\alpha-1}) j$.

Figure 4: Algorithm for computing the index of a sequence in $\mathcal{U}_{x^n}$.

respective UTCs are (recursively) obtained, as are the index of $w(y^n)$ in $\mathcal{T}[c_0^{\alpha-1}]$ and of the tail $\mathbf{t}_y$ among possible tail choices for sequences in $\mathcal{U}_{x^n}$. The function $J_{x^n}(y^n)$ is computed by composing the obtained indices using a *mixed radix number system* [34, Sec. 4.1], with the mixed radix vector $(\tau(x), M(c_0^{\alpha-1}), u_0, u_1, \ldots, u_{\alpha-1})$, where $u_a = |\mathcal{U}_{x^n[a]}|$, $a \in \mathcal{A}$. Clearly, the computational complexity of Algorithm E is determined by the computational complexity of the primitive $F[c_0^{\alpha-1}]$, and it can be readily verified that the recursion preserves polynomial complexity. The computation of the inverse function $J_{x^n}^{-1}$ is a straightforward reversal of Algorithm E, and is omitted.

The enumeration functions J_{x^n} and $J_{x^n}^{-1}$ provide a way to select a random sequence from $\mathcal{U}_{x^n}$ with close to uniform probability. The procedure consists of picking a random index in the range $0 \le j < |\mathcal{U}_{x^n}|$, and selecting $y^n = J_{x^n}^{-1}(j)$. If the index j is chosen with uniform probability, the so will be y^n in $\mathcal{U}_{x^n}$. A simple way to obtain a nearly-uniformly distributed integer in the desired range is to draw a sequence b^K of K *purely random* bits b_i (outcomes of independent fair coin tosses), for $K \ge \lceil \log |\mathcal{U}_{x^n}| \rceil$, interpret the sequence as a K-digit binary number, and let $j = b^K \mod |\mathcal{U}_{x^n}|$. However, unless $|\mathcal{U}_{x^n}|$ is a power of two, the resulting distribution will not be uniform, since some residues modulo $|\mathcal{U}_{x^n}|$ will be hit more often than others. It is possible to approach uniformity by increasing the "excess" $d = K - \lceil \log |\mathcal{U}_{x^n}| \rceil$. A natural measure of deviation from uniformity is the difference between the entropy of the actual distribution obtained on $\mathcal{U}_{x^n}$, and the entropy $\mathcal{L}_{\mathcal{U}}(x^n) = \log |\mathcal{U}_{x^n}|$ of a uniform distribution. Let Y^n denote the random variable representing the outcome of the random selection procedure outlined above. It follows from the analysis of a similar situation in [5], applied to our setting, that

$$\mathcal{L}_{\mathcal{U}}(x^n) - H(Y^n) = O\left(\exp(\lceil \log |\mathcal{U}_{x^n}| \rceil - K)\right).$$

Hence, the loss in entropy decreases exponentially with the number of "excess" random bits (notice that the entropy difference is unnormalized). Thus, the procedure is *entropy efficient*: the number

ALGORITHM U.
Input: Sequence x^n, parsing tree $T = T_{x^n}$.
Output: Sequence y^n, drawn with uniform probability from $\mathcal{U}_{x^n}$.

1. Mark all nodes of T as *unused*,
   initialize $U(v) = c_{\mathbf{p}}$ for all $\mathbf{p} \in T$.
2. Set $\mathbf{p} \leftarrow \lambda$. If $U(\mathbf{p}) = 0$, go to Step 5.
3. If $\mathbf{p}$ is *unused*,
       output $\mathbf{p}$ as the next phrase of y^n,
       mark $\mathbf{p}$ as *used*, set $U(\mathbf{p}) \leftarrow U(\mathbf{p}) - 1$,
       go to Step 2.
   End If.
4. Draw a random symbol $a \in \mathcal{A}$ with distribution $\mathrm{Prob}(a = b) = \frac{U(\mathbf{p}b)}{U(\mathbf{p})}$, $b \in \mathcal{A}$,
   set $U(\mathbf{p}) \leftarrow U(\mathbf{p}) - 1$,
   set $\mathbf{p} \leftarrow \mathbf{p}a$, and go to Step 3.
5. Pick, uniformly, a random phrase of length $|\mathbf{t}_x|$, as the tail of y^n. **Stop.**

Figure 5: Algorithm for drawing a random sequence from $\mathcal{U}_{x^n}$

K of purely random bits consumed is close to the entropy of the random variable produced, which, in turn, is close to the maximum achievable entropy for a random draw from $\mathcal{U}_{x^n}$. Upon normalization, the three resulting rates converge.

Next, we present an alternative procedure for random selection from the UTC, which can attain, in principle, a perfectly uniform distribution for the output sequence at the cost of uncertainty in the number of random bits consumed, which nevertheless will average to $\log |\mathcal{U}_{x^n}| + O(1)$.

5.2 A uniform random selection algorithm

Algorithm U in Figure 5 draws a random sequence from $\mathcal{U}_{x^n}$. In the algorithm, we assume that T_{x^n} is given, we mark nodes as *used* or *unused*, and we denote by $U(\mathbf{p})$ the number of currently *unused* nodes in the subtree rooted at $\mathbf{p} \in T_{x^n}$. We also define $U(\mathbf{p}a) = 0$ for all $\mathbf{p} \in T_{x^n}$ and $a \in \mathcal{A}$ such that $\mathbf{p}a \notin T_{x^n}$. To estimate running time, we will assume that each operation in Steps 2–4 of Algorithm U can be executed in constant time. This assumption will be elaborated on later in the section. Notice that the initializations in Step 1 can be performed in linear time, and Step 5 is executed only once.

Lemma 9 *Algorithm U terminates and outputs a sequence from $\mathcal{U}_{x^n}$. Moreover, Step 3 is executed $n + c$ times, and Step 4 is executed n times. Thus, the running time of the algorithm is $O(n)$.*

Proof. First, we observe that the output y^n of the algorithm is a concatenation of phrases from T_{x^n}, and that a phrase is marked as *used* after being appended to the output in Step 3, so it is not

output more than once. Also, since the loop in Steps 3–4 traverses the tree from the root down, a phrase is output only if all of its ancestors were found *used*. Therefore, the prefix order is respected for output phrases. For a node $\mathbf{p}$, the counter $U(\mathbf{p})$ is initialized to $c_{\mathbf{p}}$, and it is decremented every time the node is visited. Since a node is visited with positive probability only if it has an *unused* descendant, such a visit eventually leads to the emission of a phrase descending from $\mathbf{p}$, at which time the state of the emitted phrase switches from *unused* to *used*. Hence, each time execution returns to Step 2, $U(\lambda)$ has decreased by one, and $U(\mathbf{p})$ has an accurate count of unused phrases descending from $\mathbf{p}$ for all $\mathbf{p} \in T_{x^n}$. After c executions of Step 2, the algorithm eventually reaches Step 5, with all the phrases in T_{x^n} having been used and emitted. At that point, the algorithm emits a valid tail and terminates. Hence, the output y^n is a valid sequence in $\mathcal{U}_{x^n}$.

It follows from the preceding discussion that a node $\mathbf{p}$ is visited in Step 3 exactly $c_{\mathbf{p}}$ times, where, as before, $c_{\mathbf{p}}$ denotes the size of the subtree rooted at $\mathbf{p}$. Therefore, the total number of executions of Step 3 is $\sum_{\mathbf{p}\in T_{x^n}} c_{\mathbf{p}}$, which, by (17), is equal to $n+c$. Since all executions of Step 4, and all executions of Step 2 except the first one, follow an execution of Step 3, the total running time is $O(n+c) = O(n)$, as claimed (recall that Step 1 runs in linear time). Of the $c_{\mathbf{p}}$ visits to node $\mathbf{p}$, the first one (when the node is found *unused*) results in the emission of the phrase $\mathbf{p}$, while the remaining $c_{\mathbf{p}}-1$ lead to Step 4. Hence, the total number of executions of Step 4 is $\sum_{\mathbf{p}\in T_{x^n}}(c_{\mathbf{p}}-1) = n$. □

Lemma 10 *Algorithm U outputs a sequence drawn with uniform probability from* $\mathcal{U}_{x^n}$.

Proof. Let $a_1, a_2, \ldots, a_n$ denote the random symbols drawn, respectively, in the n executions of Step 4, and let $Q_i(a|a_1^{i-1})$ denote the distribution used to draw a_i. Clearly, different sequences of outcomes a^n lead to different output sequences y^n (since different choices in Step 4 lead to the emission of phrases from different subtrees). Therefore, denoting by Y^n the random variable output by the algorithm, and by A^n the sequence of random draws in Step 4, we have, for the output sequence y^n,

$$\mathrm{Prob}(Y^n = y^n) = \mathrm{Prob}(A^n = a^n) = \prod_{i=1}^{n} Q_i(a_i|a_1^{i-1})\,. \tag{34}$$

The conditional distribution Q_i depends on past draws through the state of the algorithm, i.e., the current node being visited, and the state of the counts $U(\cdot)$. We prove that y^n is emitted with uniform probability by induction on $c = |T_{x^n}|$, and assuming, initially, that $\mathbf{t}_x = \lambda$. It is readily verified that if $c = 1$, the algorithm "outputs" $y^n = \lambda$ with probability one, as expected. Assume now that $c > 1$. We monitor executions of Step 4 when $\mathbf{p} = \lambda$, which we refer to as Step 4λ. By the discussion in Lemma 9, Step 4λ is executed exactly c times. Let $i_1, i_2, \ldots, i_c$ denote the indices of these executions among the n executions of Step 4. We observe that, by the test in Step 2, Step 4λ is always reached with $U(\lambda) > 0$. The first time the step is executed, every node of the tree except the root is *unused*, and a symbol a is drawn according to the distribution $Q_1(a = b) = c_b/(c-1),\ b \in \mathcal{A}$.

Once the path to node a is taken, the algorithm will descend down the subtree $T^{(a)}$, and the iteration continues until an *unused* node is found, at which time the corresponding phrase is emitted, and the node is marked *used*. Thus, if the algorithm does not terminate, the next time Step 4λ is reached, the counts $U(\lambda)$ and $U(a)$ will have decreased by one, while the counts $U(b)$, $b \in \mathcal{A} \setminus \{a\}$, will have remained unchanged. Let $Q_{i_j}(a_{i_j}|a_1^{i_j-1}) = U_j(a_{i_j})/U_j(\lambda)$ be the probability of the symbol randomly drawn the jth time the algorithm is at Step 4λ, $1 \le j \le c$. It follows from the previous discussion that $U_j(\lambda)$ will assume the values $c-j$, $1 \le j < c$, and $U_j(a_{i_j})$, over time, will assume all the values $c_a + 1 - k$, $1 \le k \le c_a$, for all $a \in \mathcal{A}$. Hence, the multiplicative contribution of Step 4λ to the probability of the output sequence y^n is

$$P^{(\lambda)} = \prod_{j=1}^{c} Q_{i_j}(a_{i_j}|a_1^{i_j-1}) = \frac{c_0!\, c_1! \ldots c_{\alpha-1}!}{(c-1)!} = M(c_0^{\alpha-1})^{-1}.$$

Let $a \in \mathcal{A}$ be a fixed child of the root. As mentioned, following each visit to a, the algorithm will emit a phrase from the subtree rooted at a, before returning to the root λ. If we remove the initial symbol a from each of these phrases, their concatenation forms the sequence $y^n[a] \in \mathcal{U}(T^{(a)})$ previously defined in Section 3.1. Moreover, if we ignore anything that happens when other branches from λ are taken, then the sequence of steps taken while the algorithm is visiting nodes in $T^{(a)}$ forms a complete execution of the algorithm on that tree. Since $c_a < c$, the induction hypothesis holds, and we have $\mathrm{Prob}(Y^n[a] = y^n[a]) = |\mathcal{U}(T^{(a)})|^{-1}$. The same reasoning applies, independently, to each child of the root. Thus, for the output sequence y^n, we have

$$\mathrm{Prob}(Y^n = y^n) = P^{(\lambda)} \prod_{a \in \mathcal{A}} \mathrm{Prob}(Y^n[a] = y^n[a]) = M(c_0^{\alpha-1})^{-1} \prod_{a \in \mathcal{A}} |\mathcal{U}(T^{(a)})|^{-1} = |\mathcal{U}_{x^n}|^{-1}\,,$$

completing the proof of the claim when $\mathbf{t}_x = \lambda$. When $|\mathbf{t}_x| > 0$, the uniform random choice of a tail in Step 5 preserves the uniform probability of y^n. □

We now discuss in more detail the mechanics of the random draws in Step 4 of Algorithm U, for which the assumption of constant execution time might be arguable. The issue is important not only for its impact on the complexity of the algorithm, but also because it determines the total amount of randomness the algorithm requires to produce its output. We assume, for simplicity, that $\mathbf{t}_x = \lambda$.

Assume the algorithm has access to a semi-infinite sequence b^∞ of purely random bits. The stream of random bits could be used to produce each random draw independently, using the classical technique of [8] (see also [3, Sec. 5.12]). Denoting by N_i the number of purely random bits required to generate the random variable a_i with distribution Q_i, it follows from the results of [8] that

$$EN_i \le H(Q_i) + 2 \le \log \alpha + 2. \tag{35}$$

The execution time of the procedure is proportional to the number of purely random bits consumed, so the *expected* running time per random draw is indeed constant. Notice also that since the largest

denominator of a probability value in Step 4 is $c-1$, the operations required for the random draw require registers of size $O(\log c)$, which is in line with the other operations, and can reasonably be assumed to be operated on in constant time. However, the number of purely random bits (and execution time) required in the worst case is unbounded. In fact, any algorithm that uses fair coins to generate random variables with probabilities that cannot be written as sums of dyadic fractions must have unbounded execution paths. In our case, the probabilities in the distributions Q_i have denominators that run over a range of consecutive integers, so they will generally not be expressible as sums of dyadic fractions.

Aside from the issue of unbounded execution paths, which is inevitable if exact probabilities are desired in the random draws, the above procedure is not entropy-efficient, as it handles each random draw independently, and each draw incurs a constant redundancy over the entropy of its output. We now describe a more efficient procedure, which will be asymptotically optimal in random bit consumption, as was the case with the procedure in Section 5.1.

The random drawing procedure will be based on a decoder for an *Elias code* [35] (also referred to as a *Shannon-Fano-Elias* code [3]), a precursor of the *arithmetic code* [36, 37]. We will obtain the sequence a^n by feeding the random sequence b^∞ to an Elias decoder, and driving the decoder with the distributions Q_i, i.e., a first symbol $a_1 \in \mathcal{A}$ is produced by the decoder using the distribution Q_1, the second symbol uses the distribution Q_2 and so on, until a_n is produced. This general procedure for generating arbitrarily-distributed discrete random sequences from fair coin tosses is described and analyzed in [11] under the name *interval algorithm*. The procedure generalizes the results of [8], and provides an efficient sequential implementation for generating discrete random sequences from fair coin tosses (or, as a matter of fact, from arbitrary coins). Our setting corresponds to the iterative variant of the algorithm in [11], with a different target distribution used at each decoding step.[8] Let N_R denote the length of the prefix of b^∞ required to generate a^n with the distribution in (34). It follows from the results of [11] that, in analogy with (35), we have

$$EN_R \leq H(A^n) + 3. \tag{36}$$

By (34) and the result of Lemma 10, we have $H(A^n) = H(Y^n) = \mathcal{L}_{\mathcal{U}}(x^n)$, and, therefore,

$$EN_R \leq \mathcal{L}_{\mathcal{U}}(x^n) + 3.$$

On the other hand, given the tree T_{x^n}, y^n is obtained deterministically from the sequence b^{N_R}. Therefore, we must have

$$EN_R \geq H(Y^n) = \mathcal{L}_{\mathcal{U}}(x^n).$$

Thus, the expected random bit consumption of the Elias decoding procedure is optimal up to an additive constant. We summarize the foregoing discussion in the following theorem, where we assume that Algorithm U incorporates the Elias decoding procedure for the random draws in Step 4.

[8] Of course, using distributions that change as symbols are decoded is routine practice in the context-driven arithmetic decoders used in universal data compression [38]—here, we refer to the specific results of [11] on using Elias decoders as random number generators.

Theorem 4 *Algorithm U outputs a sequence y^n uniformly distributed in $\mathcal{U}_{x^n}$. The expected number of purely random bits required per sample is $\overline{\mathcal{L}}_{\mathcal{U}}(x^n) + O(n^{-1})$, which is asymptotically optimal.*

Proof. The claims on y^n and the expected number of purely random bits were established in Lemmas 9 and 10, and in the discussion preceding the theorem. □

Although the expected number of purely random bits in Algorithm U is asymptotically optimal, as mentioned, some sample paths will require an unbounded number of random bits (and execution time). Furthermore, the Elias code requires arithmetic operations with registers of length $O(n)$ in the worst case, for which an assumption of constant execution time might be unreasonable. The register length problems of Elias coding were solved with the emergence of arithmetic coding [36, 37]. An arithmetic code with bounded-length registers, however, incurs a certain redundancy (see also [39]), which would translate to a departure from a perfectly uniform distribution in our application. The trade-off is analogous to that discussed for the enumerative procedure of Section 5.1. In fact, a careful analysis of both procedures reveals that they perform essentially the same computation, and Algorithm U can be regarded as just an alternative, more efficient implementation of a random selection procedure based on Algorithm E.

6 Universal simulation of individual sequences

Informally, given an individual sequence x^n, we are interested in producing a (possibly random) "simulated" sequence y^n with the following properties:

S1. y^n is statistically similar to x^n;

S2. given that y^n satisfies Condition S1, there is as much uncertainty in the choice of y^n as possible.

Condition S1 is initially stated in a purposely vague fashion, as the desired similarity criterion may vary from setting to setting. In [5], for example, x^n is an individual sequence assumed to have been emitted by a source from a certain parametric class, and a strict criterion is used, where y^n must obey exactly the same (unknown) probability law, from the parametric class, as x^n. Other works on simulation of random processes (e.g., [9, 11]) assume full knowledge of the source being simulated, and do not use an individual sequence x^n as part of the input.

Condition S2 is desirable to avoid, in the extreme case, a situation where the simulator deterministically outputs a copy of x^n (which certainly satisfies Condition S1 for any reasonable definition of "similarity") as its own "simulation." We wish to have as much variety as possible in the space of possible simulations of x^n.

We now formalize our notion of sequence simulation, propose a simulation scheme based on universal types, and prove that it is, in a well defined mathematical sense, optimal with respect to Conditions S1 and S2, even when allowing competing schemes to satisfy weaker conditions.

A *simulation scheme* for individual sequences is a function $\mathcal{S}$ that maps sequences x^n, for all n, to random variables $Y^n = \mathcal{S}(x^n)$ taking values on $\mathcal{A}^n$. We say that $\mathcal{S}$ is *faithful* to $\{x^n\}$ if for every integer $k \geq 1$, and positive real number ε, we have

$$\text{Prob}_{Y^n}\left\{ \|\hat{P}^{(k)}_{x^n} - \hat{P}^{(k)}_{Y^n}\|_\infty \leq \varepsilon \right\} \to 1 \quad \text{as} \;\; n \to \infty. \tag{37}$$

Faithfulness will be our criterion for similarity of sequence statistics in Condition S1. The criterion is, in the realm of empirical probabilities, of the same flavor as that used in [9] for random processes. To satisfy Condition S2, we will seek simulation schemes that maximize the entropy of $\mathcal{S}(x^n)$ given that $\mathcal{S}$ is faithful to $\{x^n\}$. This can be seen as analogous to the criterion used in [5], where a minimization of the mutual information between the input random source and the simulated output is sought, given that the output satisfies the probability law of the input.

The random selection algorithms of Sections 5.1 and 5.2 suggest, very naturally, a simulation scheme $\mathcal{S}_{\mathcal{U}}$, where $\mathcal{S}_{\mathcal{U}}(x^n)$ is a random sequence Y^n uniformly distributed on $\mathcal{U}_{x^n}$.

Theorem 5 *The simulation scheme $\mathcal{S}_{\mathcal{U}}$ is faithful to all $\{x^n\}$. Its output sequence has entropy rate $\mathbf{H}(\mathcal{S}_{\mathcal{U}}(x^n)) = \overline{\mathcal{L}}_{\mathcal{U}}(x^n)$.*

Proof. The faithfulness of the scheme is a direct consequence of Theorem 1. In fact, $\mathcal{S}_{\mathcal{U}}$ satisfies (37) with probability *equal* to one for every sufficiently large n; i.e., it is faithful *everywhere*, which will not be a requirement for other schemes we will compare $\mathcal{S}_{\mathcal{U}}$ against. $\mathcal{S}_{\mathcal{U}}$ selects an output sequence uniformly from $\mathcal{U}_{x^n}$, so, by the definition of $\overline{\mathcal{L}}_{\mathcal{U}}(x^n)$ in (29), its entropy is as claimed. □

Lemma 11 *Let $\{x^n\}$ be a sequence such that*

$$\lim_{k\to\infty} \overline{\lim}_n \left| \overline{\mathcal{L}}_{LZ}(x^n) - \hat{\mathbf{H}}_k(x^n) \right| = 0, \tag{38}$$

and let $\mathcal{S}$ be a simulation scheme satisfying

$$\mathbf{H}\left(\mathcal{S}(x^n)\right) \geq \overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \Delta \tag{39}$$

for some $\Delta > 0$, and all sufficiently large n. Then, $\mathcal{S}$ is not faithful to $\{x^n\}$.

Proof. By the hypothesis of the lemma on $\{x^n\}$, there exists an integer K such that for an arbitrarily chosen $\delta_1 > 0$, and all $k \geq K$, we have

$$\overline{\lim}_n \left| \overline{\mathcal{L}}_{\text{LZ}}(x^n) - \hat{\mathbf{H}}_k(x^n) \right| \leq \delta_1. \tag{40}$$

Combining with the result of Corollary 4, we further obtain

$$\left| \overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \hat{\mathbf{H}}_k(x^n) \right| \leq \delta_1 + \delta_2, \tag{41}$$

for sufficiently large n and an arbitrary $\delta_2 > 0$. Let y^n be any sequence in $\mathcal{A}^n$. From the universality of the Lempel-Ziv code in the class of finite-memory sources [6, 3], we have

$$\overline{\mathcal{L}}_{\mathrm{LZ}}(y^n) \leq \hat{\mathbf{H}}_k(y^n) + \delta_3\,, \tag{42}$$

uniformly in y^n for any given k, sufficiently large n, and arbitrary $\delta_3 > 0$. Define $\varepsilon_k(y^n) = |\hat{\mathbf{H}}_k(y^n) - \hat{\mathbf{H}}_k(x^n)|$. Using this definition together with (41) and (42), we obtain

$$\overline{\mathcal{L}}_{\mathrm{LZ}}(y^n) \leq \overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \varepsilon_k(y^n) + \delta_1 + \delta_2 + \delta_3\,, \tag{43}$$

for all $k \geq K$ and sufficiently large n. Choose $\delta_1 = \delta_2 = \delta_3 = \Delta/8$, and let $\mathcal{Y}_{n,k} = \{y^n \in \mathcal{A}^n : \varepsilon_k(y^n) \leq \Delta/8\,\}$. Then, (43) yields

$$\overline{\mathcal{L}}_{\mathrm{LZ}}(y^n) \leq \overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \Delta/2, \quad y^n \in \mathcal{Y}_{n,k}\,.$$

We define the following lossless binary encoding for sequences $y^n \in \mathcal{A}^n$: if $y^n \in \mathcal{Y}_{n,k}$, encode y^n by a symbol 0 followed by the LZ'78 binary encoding of y^n; otherwise, encode y^n by a symbol 1 followed by the binary representation, of length $\lceil n \log \alpha \rceil$, of y^n interpreted as a number in base α. Let $p_{n,k} = \mathrm{Prob}_{\mathcal{S}}\,\{y^n \in \mathcal{Y}_{n,k}\}$. Then, the expected normalized code length of the encoding under $\mathrm{Prob}_{\mathcal{S}}$ is

$$\overline{\mathcal{L}}_{\mathcal{S}} = (\overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \Delta/2)p_{n,k} + (\log \alpha)(1 - p_{n,k}) + o(1). \tag{44}$$

By Shannon's fundamental entropy bound, and the assumptions of the lemma, we must have

$$\overline{\mathcal{L}}_{\mathcal{S}} \geq \mathbf{H}\,(\mathcal{S}(x^n)) \geq \overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \Delta\,. \tag{45}$$

Combining (44) and (45), and choosing n sufficiently large so that, additionally, the $o(1)$ term in (44) does not exceed $\Delta/4$, we obtain

$$(\overline{\mathcal{L}}_{\mathcal{U}}(x^n) + \Delta/2)p_{n,k} + (\log \alpha)(1 - p_{n,k}) \geq \overline{\mathcal{L}}_{\mathcal{U}}(x^n) + 3\Delta/4\,,$$

or, rearranging terms,

$$p_{n,k} \leq \frac{\log \alpha - \overline{\mathcal{L}}_{\mathcal{U}}(x^n) - 3\Delta/4}{\log \alpha - \overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \Delta/2} < 1\,, \tag{46}$$

where the last inequality follows from the fact that $\mathbf{H}(\mathcal{S}(x)) \leq \log \alpha$, and, thus, by (45), $\log \alpha - \overline{\mathcal{L}}_{\mathcal{U}}(x^n) - \Delta \geq 0$, and both the numerator and the denominator of the fraction in (46) are positive. Now, let δ be a positive real number satisfying

$$\alpha^k(\alpha + 1)\delta \log \delta^{-1} \leq \min\{\,\frac{\Delta}{16}, \frac{1}{2\alpha^{k+1}}\,\}, \tag{47}$$

and choose n sufficiently large so that the term $O(\log n/n)$ in Lemma 4 does not exceed $\Delta/16$. Then, it follows from Lemma 4, (47), (46), and the definition of $\mathcal{Y}_{n,k}$, that for $k \geq K$ and sufficiently large n, we have

$$\begin{aligned}
&\mathrm{Prob}_{\mathcal{S}}\left\{\|\hat{P}^{(k)}_{x^n} - \hat{P}^{(k)}_{y^n}\|_\infty \leq \delta \text{ and } \|\hat{P}^{(k+1)}_{x^n} - \hat{P}^{(k+1)}_{y^n}\|_\infty \leq \delta\right\} \\
&\quad \leq \mathrm{Prob}_{\mathcal{S}}\left\{|\hat{\mathbf{H}}_k(x^n) - \hat{\mathbf{H}}_k(y^n)| \leq \frac{\Delta}{8}\right\} \;\leq\; \mathrm{Prob}_{\mathcal{S}}\,\{y^n \in \mathcal{Y}_{n,k}\} \;=\; p_{n,k} \;<\; 1\,.
\end{aligned}$$

Therefore, $\mathcal{S}$ is not faithful to $\{x^n\}$. □

The proof of Lemma 11 relies on the assumption (38). Sequences that do not satisfy the assumption exist, as shown in the following example.

Example 7. Let m be a positive integer, and let C_m denote a binary counting sequence of order m (see Example 4). A binary *de Bruijn* sequence of order m [40], denoted by $B_m = b_0^{2^m-1}$, has the following property: $N(u^m, b_0^{2^m+m-2}) = 1$ for all $u^m \in \{0,1\}^m$, where $b_0^{2^m-1}$ is extended cyclically to length 2^m+m-1. De Bruijn sequences exist for all orders m, and have been extensively studied (see, e.g., [41, 42]). The sequence x^{n_m} is defined in the following equation, where we use (non-commutative) multiplicative notation for string concatenation, and indices into B_m are computed modulo 2^m.

$$x^{n_m} = C_m \prod_{i=0}^{2^m-1} \prod_{j=m+1}^{2^m} b_i^{i+j-1}, \tag{48}$$

In words, x^{n_m} consists of a counting sequence C_m followed by the concatenation of all the prefixes of length $j \geq m+1$, in increasing length order, of each of the 2^m cyclic shifts of B_m. The parsing tree of x^{n_m} consists of a complete balanced tree of depth m, where from each leaf $u^m \in \{0,1\}^m$ "hangs" a chain of nodes of degree one corresponding to the completion, of length $2^m - m$, of the cyclic shift of B_m starting with u^m. The strings b_i^{i+j-1} in (48), together with the phrases of C_m, comprise all the phrases of x^{n_m}, as each string uniquely extends a previous phrase by one symbol.

Lemma 12 *We have* $\lim_{m\to\infty} \hat{\mathbf{H}}_k(x^{n_m}) = 1$ *for all* $k \geq 0$, *and* $\lim_{m\to\infty} \overline{\mathcal{L}}_{LZ}(x^{n_m}) = 0$.

Proof. First, it is established by direct computation that the length and number of phrases of x^{n_m} are, respectively, $n_m \approx 2^{3m}$, and $c_m \approx 2^{2m}$, up to lower order terms. Thus, we have $\overline{\mathcal{L}}_{\mathrm{LZ}}(x^{n_m}) \approx (c_m \log c_m)/n_m \to 0$ as $m \to \infty$. We claim that the distribution of k-tuples in x^{n_m} is nearly uniform for all $k \leq m$. Let u^m be a binary m-tuple, and let $s(u)$ be the unique index in the range $0 \leq s(u) < 2^m$ such that $b_{s(u)}^{s(u)+m-1} = u^m$. For each phrase length t, $m \leq t \leq 2^m$, u^m occurs exactly once in each phrase of the form $b_{s'}^{s'+t-1}$, $s(u) \leq s' \leq s(u)+t-m$, i.e., in $t-m+1$ phrases, independently of u^m. Therefore, m-tuples that are fully contained in phrases of length $t \geq m$ are uniformly distributed, and, consequently, so are other k-tuples for $k < m$ (it is easily verified that phrases of length less than m, which occur in C_m, are also uniformly distributed). By an argument similar to the one used in the proof of Theorem 1, k-tuples that are not fully contained in phrases occur in negligible proportion. It follows that the distribution of k-tuples in x^{n_m} approaches uniformity as $m \to \infty$ for every fixed $k \geq 1$. Thus, $\hat{\mathbf{H}}_{k-1}(x^{n_m}) \to 1$ as $m \to \infty$, for every $k \geq 1$. □

Although the example of $\{x^{n_m}\}$ was presented in the "sequence of sequences" setting, it is also readily verified that the concatenation $\prod_{m=1}^{\infty} x^{n_m}$ gives one semi-infinite sequence x^∞ that has LZ78 compressibility [6] equal to zero, but finite-memory compressibility [3] (and, thus, also *finite-state* compressibility [6, 17]) equal to one. The example is of the same flavor as examples of sequences

that differentiate between various finite- and infinite-memory variants of Lempel-Ziv compression, presented in [43] and [44]. In particular, an example presented in [43] is compressible to rate 1/2 by the LZ77 variant [45], but it is incompressible by finite-state compressors; an example presented in [44] differentiates between LZ77 and LZ78, and is also based on de Bruijn sequences.

Example 7 highlights the fact that the "raw" LZ78 incremental parsing captures more than just the finite-memory patterns of the data being parsed. In cases like that of the example, the type class $\mathcal{U}_{x^{n_m}}$ will be fairly small, and will not include sequences that would be considered statistically indistinguishable from x^{n_m} judging by their finite-memory properties alone. This "anomaly" is avoided in the original paper [6] by dividing x^n into blocks of length N, and carrying out the incremental parsing and encoding independently for each block. Compressibility is then defined by sending the number of blocks to infinity first, and then the block size, in analogy to the way finite-state compressibility is defined.

Universal types could also be defined as based on a "blocked" incremental parsing, which would avoid the need for the condition on $\{x^n\}$ in Lemma 11. However, the modification would add complexity to the notation and analysis, without adding significant insight. The "blocked" UTCs would be Cartesian products of the simple UTCs described here, for sub-blocks of length N of x^n. We note also that most of the literature on LZ78 (see, e.g., [3, 46, 32, 31] for a sample) focuses on the "raw" incremental parsing without blocking. In any case, the "anomaly" applies to very few sequences, as shown in the following lemma.

Lemma 13 *The condition (38) holds for all x^n except for a subset $\mathcal{X}_n \in \mathcal{A}^n$ whose volume vanishes as $n \to \infty$, under any stationary-ergodic measure on x^n.*

Proof. Let P_X be any stationary ergodic measure that applies to x^n. It follows from the sample converse theorems for source coding in [47, 48] that

$$\underline{\lim}_n \overline{\mathcal{L}}_{\rm LZ}(x^n) \geq \mathbf{H}(X), \quad \text{a.s.}, \tag{49}$$

On the other hand, from the universality of the LZ code, we have [3, Sec. 12.10]

$$\overline{\lim}_n \overline{\mathcal{L}}_{\rm LZ}(x^n) \leq \mathbf{H}(X), \quad \text{a.s.},$$

and from the convergence of the kth order empirical entropy for stationary ergodic processes,

$$\lim_k \lim_n \hat{\mathbf{H}}_k(x^n) = \mathbf{H}(X), \quad \text{a.s.}$$

Putting these together, we obtain

$$\lim_k \lim_n |\overline{\mathcal{L}}_{\rm LZ}(x^n) - \hat{\mathbf{H}}_k(x^n)| = 0, \quad \text{a.s.}.$$

Since a.s. convergence implies convergence in probability, for any $\varepsilon > 0$, there exists an integer K such that for all $k \geq K$,

$$P_X \left\{ |\overline{\mathcal{L}}_{\rm LZ}(x^n) - \hat{\mathbf{H}}_k(x^n)| < \varepsilon \right\} \to 0 \ \text{ as } \ n \to \infty \, .$$

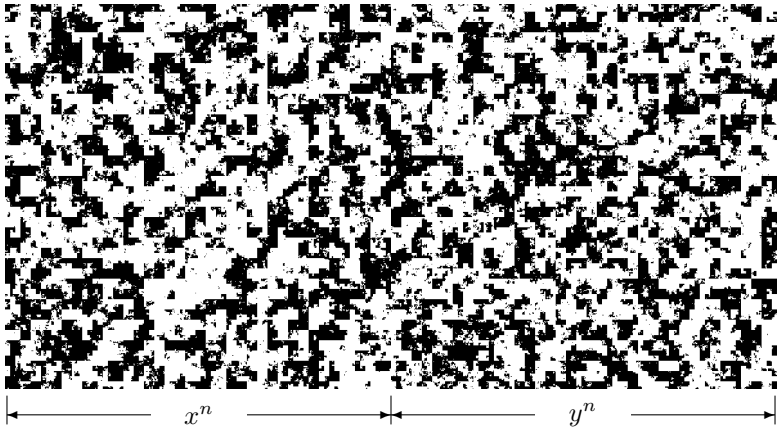

Figure 6: Texture simulation

□

Notice that although a probability measure on input sequences x^n is used in Lemma 13, it just provides a formal way of saying that "most" individual sequences satisfy the condition of Lemma 11. All the results still apply to individual sequences, and the proposed simulation scheme $\mathcal{S}_{\mathcal{U}}$ makes no stochastic assumptions on x^n.

We summarize the discussion in the following theorem, which establishes the optimality of $\mathcal{S}_{\mathcal{U}}$, in the sense that no scheme that complies with Condition S1 can do significantly better than $\mathcal{S}_{\mathcal{U}}$ on Condition S2.

Theorem 6 *Every faithful simulation scheme $\mathcal{S}$ must satisfy*

$$\mathbf{H}(\mathcal{S}(x^n)) \leq \mathbf{H}(\mathcal{S}_{\mathcal{U}}(x^n)) + \Delta$$

for all $\Delta > 0$, sufficiently large n, and all sequences x^n, except for a subset of vanishing volume as $n \to \infty$, under any stationary ergodic measure on these sequences.

The simulation scheme $\mathcal{S}_{\mathcal{U}}$ was tested on some binary textures. For the example in Figure 6, a 1024×1024 binary texture was generated, and scanned with a Peano plane-filling scan, to produce a binary sequence x^n of $n = 2^{20}$ samples. The sequence x^n was then "simulated" by generating a uniform random sample y^n from $\mathcal{U}_{x^n}$. Finally, the sequence y^n was mapped back, reversing the same Peano scan order, to a 1024×1024 image. The left half of Figure 6 shows a 600×600 patch of the texture x^n, while the right half shows a 600×600 patch of the simulation y^n (the smaller patches are used to fit the page without sub-sampling or altering the visual quality of the images). It is evident from the figure that the right half indeed "looks like" the left half, and the seam between the images is unnoticeable. Yet, the right half is completely different from the left half, and was selected from a very large class of possible simulation images. In fact, the size of $\mathcal{U}_{x^n}$ in this example was estimated using the recursion (14), resulting in $\log |\mathcal{U}_{x^n}| \approx 109,700$.

Acknowledgments. Thanks to Neri Merhav, Erik Ordentlich, Wojciech Szpankowski, Alfredo Viola, Marcelo Weinberger, and Tsachy Weissman for very useful discussions. Thanks also to Boris Pittel for pointing out the connection between the size of a universal type class and labeling of trees.

References

[1] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. New York: Academic, 1981. 1, 2, 9

[2] I. Csiszár, "The method of types," *IEEE Trans. Inform. Theory*, vol. IT-44, no. 6, pp. 2505–2523, Oct. 1998. 1, 2, 18

[3] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, Inc., 1991. 1, 2, 3, 7, 9, 18, 24, 25, 28, 29, 30

[4] E. L. Lehmann, *Theory of point estimation*. New York: Wiley, 1983. 2

[5] N. Merhav and M. J. Weinberger, "On universal simulation of information sources using training data," *IEEE Trans. Inform. Theory*, vol. 50, no. 1, pp. 5–20, Jan. 2004. 2, 4, 21, 26, 27

[6] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 530–536, Sept. 1978. 2, 7, 17, 28, 29, 30

[7] J. Rissanen, "A universal data compression system," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 656–664, Sept. 1983. 2

[8] D. E. Knuth and A. Yao, "The complexity of nonuniform random number generation," in *Algorithms and Complexity, New Directions and Results*, J. F. Traub, Ed. New York: Academic Press, 1976, pp. 357–428. 4, 24, 25

[9] T. S. Han and S. Verdú, "Approximation theory of output statistics," *IEEE Trans. Inform. Theory*, vol. 39, pp. 752–772, May 1993. 4, 26, 27

[10] Y. Steinberg and S. Verdú, "Simulation of random processes and rate-distortion theory," *IEEE Trans. Inform. Theory*, vol. 42, no. 1, pp. 63–86, Jan. 1996. 4

[11] T. S. Han and M. Hoshi, "Interval algorithm for random number generation," *IEEE Trans. Inform. Theory*, vol. 43, no. 2, pp. 599–611, Mar. 1997. 4, 25, 26

[12] I. Csiszár and P. C. Shields, "Redundancy rates for renewal and other processes," *IEEE Trans. Inform. Theory*, vol. 42, no. 6, pp. 2065–2072, Nov. 1996. 4

[13] N. Merhav, "Universal coding with minimum probability of codeword length overflow," *IEEE Trans. Inform. Theory*, vol. 37, no. 3, pp. 556–563, May 1991. 4

[14] J. Ziv and N. Merhav, "Estimating the number of states of a finite-state source," *IEEE Trans. Inform. Theory*, vol. 38, no. 1, pp. 61–65, Jan. 1992. 4

[15] G. Seroussi, "On the number of t-ary trees with a given path length," Hewlett-Packard Laboratories, Technical Report HPL-2004-127, July 2004, (submitted). [Online]. Available: http://www.hpl.hp.com/techreports/2004/HPL-2004-127.html 5, 19

[16] D. E. Knuth, *The Art of Computer Programming. Fundamental Algorithms*, 3rd ed. Reading, MA: Addison-Wesley, 1997, vol. 1. 5, 6, 7, 14, 20

[17] M. Feder, N. Merhav, and M. Gutman, "Universal prediction of individual sequences," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1258–1270, July 1992. 9, 29

[18] D. E. Knuth, *The Art of Computer Programming. Sorting and Searching*, 2nd ed. Reading, MA: Addison-Wesley, 1998, vol. 3. 10, 11, 12

[19] G. H. Gonnet and J. I. Munro, "Heaps on heaps," *SIAM J. Comput.*, vol. 15, no. 4, pp. 964–971, Nov. 1986. 11

[20] H.-K. Hwang and J.-M. Steyaert, "On the number of heaps and the cost of heap construction," in *Mathematics and computer science, II, Trends Math.* Basel: Birkhäuser, 2002, pp. 295–310. 11

[21] D. G. Champerknowne, "The construction of decimals normal in the scale of ten," *Journal of the London Math. Soc.*, vol. 8, pp. 254–260, 1933. 17

[22] N. J. A. Sloane, "The on-line encyclopedia of integer sequences," published electronically at http://www.research.att.com/ njas/sequences/. 17

[23] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*. Amsterdam: North-Holland Publishing Co., 1983. 17

[24] P. Whittle, "Some distribution and moment formulae for the Markov chain," *J. Roy. Stat. Soc., Ser. B*, vol. 17, pp. 235–242, 1955. 18

[25] L. D. Davisson, G. Longo, and A. Sgarro, "The error exponent for the noiseless encoding of finite ergodic Markov sources," *IEEE Trans. Inform. Theory*, vol. 27, no. 4, pp. 431–438, July 1981. 18

[26] P. Jacquet and W. Szpankowski, "Markov types and minimax redundancy for Markov sources," *IEEE Trans. Inform. Theory*, vol. 50, no. 7, pp. 1393–1402, July 2004. 18

[27] F. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Trans. Inform. Theory*, vol. 41, no. 3, pp. 653–664, May 1995. 18

[28] T. M. Cover, "Enumerative source encoding," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 73–77, Jan. 1973. 18, 20

[29] C. Knessl and W. Szpankowski, "Enumeration of binary trees, Lempel-Ziv78 parsings, and universal types," 2004, preprint. 19

[30] M. J. Weinberger, N. Merhav, and M. Feder, "Optimal sequential probability assignment for individual sequences," *IEEE Trans. Inform. Theory*, vol. IT-40, pp. 384–396, Mar. 1994. 20

[31] G. Louchard and W. Szpankowski, "On the average redundancy rate of the Lempel-Ziv code," *IEEE Trans. Inform. Theory*, vol. 43, no. 1, pp. 2–8, Jan. 1997. 20, 30

[32] S. A. Savari, "Redundancy of the Lempel-Ziv incremental parsing rule," *IEEE Transactions on Information Theory*, vol. 43, no. 1, pp. 9–21, Jan. 1997. 20, 30

[33] D. H. Lehmer, "The machine tools of combinatorics," in *Applied Combinatorial Mathematics*, E. F. Beckenbach, Ed. New York: Wiley, 1964, pp. 5–30. 20

[34] D. E. Knuth, *The Art of Computer Programming. Seminumerical Algorithms*, 3rd ed. Reading, MA: Addison-Wesley, 1998, vol. 2. 21

[35] F. Jelinek, *Probabilistic Information Theory.* New York: McGraw-Hill, 1968. 25

[36] J. Rissanen, "Generalized kraft inequality and arithmetic coding," *IBM J. Res. Develop.*, vol. 20, pp. 197–300, May 1976. 25, 26

[37] R. Pasco, "Source coding algorithms for fast data compression," Ph.D. Thesis, Stanford University, 1976. 25, 26

[38] J. Rissanen and G. G. Langdon, "Universal modeling and coding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 12–23, Jan. 1981. 25

[39] M. J. Weinberger, A. Lempel, and J. Ziv, "A sequential algorithm for the universal coding of finite memory sources," *IEEE Trans. Inform. Theory*, vol. 38, no. 3, pp. 1002–1014, May 1992. 26

[40] N. G. de Bruijn, "A combinatorial problem," *Koninklijke Nederlands Akademie van Wetenschappen, Proceedings*, vol. 49 Part 2, pp. 758–764, 1946. 29

[41] S. W. Golomb, *Shift Register Sequences.* San Francisco: Holden-Day, 1967. 29

[42] J. Marshall Hall, *Combinatorial Theory*, 2nd ed. New York: John Wiley & Sons, 1986. 29

[43] P. C. Shields, "Performance of LZ algorithms on individual sequences," *IEEE Trans. Inform. Theory*, vol. 45, no. 4, pp. 1283–1288, May 1999. 30

[44] L. A. Pierce II and P. Shields, "Sequences incompressible by SLZ (LZW), yet fully compressible by ULZ," in *Numbers, Information and Complexity*, I. Althöfer, N. Cai, G. Dueck, L. Khachatrian, M. Pinsker, A. Sarközy, I. Wegener, and Z. Zhang, Eds. Kluwer Academic Publishers, 2000, pp. 385–390. 30

[45] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 337–343, May 1977. 30

[46] E. Plotnik, M. J. Weinberger, and J. Ziv, "Upper bounds on the probability of sequences emitted by finite-state sources and on the redundancy of the Lempel-Ziv algorithm," *IEEE Trans. Inform. Theory*, vol. 38, no. 1, pp. 66–72, Jan. 1992. 30

[47] A. R. Barron, "Logically smooth density estimation," Ph.D. dissertation, Stanford University, Stanford, California, 1985. 30

[48] J. C. Kieffer, "Sample converses in source coding theory," *IEEE Trans. Inform. Theory*, vol. 37, no. 2, pp. 263–268, Mar. 1991. 30