Max-Planck-Institut für Mathematik in den Naturwissenschaften Leipzig

Sequential Recurrence-Based Multidimensional Universal Source Coding of Lempel-Ziv Type

by

Tyll Krüger, Guido Montúfar, Ruedi Seiler, and Rainer Siegmund-Schultze

Preprint no.: 86 2014

Sequential Recurrence-Based Multidimensional Universal Source Coding of Lempel-Ziv Type

Tyll Krueger, Guido Montúfar, Ruedi Seiler, and Rainer Siegmund-Schultze

***Abstract*—We define an algorithm that parses multidimensional arrays sequentially into mainly unrepeated but nested multidimensional sub-arrays of increasing size, and show that the resulting sub-block pointer encoder compresses almost every realization of any finite-alphabet ergodic process on $\mathbb{Z}^d_{\geq 0}$ to the entropy, in the limit.**

***Index Terms*—universal code, Lempel-Ziv algorithm, multiple recurrence, multidimensional ergodic process.**

I. Introduction

THIS paper is about the design of sequential dimension-preserving parsing and coding algorithms for multidimensional arrays of data, with optimality proofs.

The Lempel-Ziv (LZ) algorithm [1], [2] parses an infinite sequence of symbols sequentially into non-overlapping consecutive blocks, with each block corresponding to the shortest sequence of symbols that has not appeared as a previous block in the parsing. The original sequence is then expressed as a sequence of words, each of which is equal to one of the previous words plus an additional symbol. Each word is encoded by a pointer to the previously occurring sub-word plus an additional symbol. In the limit of infinite sequences, this procedure compresses almost every realization of any stationary process down to its entropy. Many lossless data compression algorithms are based on similar recurrence-based parsings and pointers.

Lempel and Ziv [3] showed that the algorithm described above can also be used to compress multidimensional data, by first transforming the data to a 1-dimensional stream. This is done by scanning the data with a space-filling Peano-Hilbert curve that preserves the local correlations. However, this approach results in an encryption of the data correlations, due to the inevitable fractal nature of the scanning curve. Furthermore, the local correlations recorded by the scanning curve are essentially enclosed in blocks of side length equal to powers of two. In this respect, the algorithm takes correlation-lengths into account that are systematically smaller than optimal. Therefore, the convergence rate of this algorithm can be expected to be sub-optimal.

T. Krueger is with the Department of Computer Science and Engineering at the Wrocław University of Technology, Wrocław, Poland. E-mail: tyll.krueger@pwr.wroc.pl.

G. Montúfar is with the Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany. E-mail: montufar@mis.mpg.de.

R. Seiler is emeritus of the Department of Mathematics at the Technische Universität Berlin, Berlin, Germany. E-mail: ruedi.seiler@integral-learning.de.

R. Siegmund-Schultze is with the Department of Mathematics at the Technische Universität Berlin, Berlin, Germany. E-mail: rainer.siegmund-schultze@integral-learning.de.

August 19, 2014.

We propose a resort that leaves the multidimensional structure of the data untouched. Our algorithm parses the data into multidimensional blocks. As in the LZ algorithm, the blocks are constructed in a way that allows to encode them by pointers to previous blocks plus some additional symbols. Here, a main difficulty is to control the number of allowed block shapes together with the amount of block overlaps. If multiple block shapes are allowed (which can be useful to avoid overlaps or to maintain the number of additional symbols small), then the shape of each block has to be encoded. In the worst case, this leads to a hopeless combinatorial explosion. On the other hand, if the blocks are allowed to overlap, then the corresponding portions of the data may need to be encoded multiple times.

Our algorithm uses cubical blocks of variable size, and controls their overlap by allowing both word repetitions and entirely new words. Although in the LZ algorithm all words are different from each other and each new word consists of a previously observed word plus a single additional symbol, repeated words or entirely new words are not a fundamental problem for achieving optimal compression. All that is needed is that, on the whole, most blocks contain large previously occurring blocks and that their size adjusts to the size of the data array and its entropy.

Regular parsings with non-overlapping cubical blocks of constant size can be used to obtain optimal codes, whereby the size of the blocks has to be chosen depending on the size of the data array and its (empirical) entropy. We discuss this approach in Section III. Such a parsing algorithm is not sequential: each time new data entries become available, the block size has to be adjusted to the new data volume and the entire code has to be recomputed.

In order to overcome this problem, it is desirable to have an algorithm that parses and encodes only the new data entries, leaving the old parts of the parsing and encoding unchanged. We achieve this by adjusting the block size only for the new data entries. We discuss this approach in Section IV. As we will show, this can be done in an asymptotically optimal way by following a simple rule: the block size is increased when the proportion of repeated words of the current parsing block size surpasses a pre-specified threshold $0 < \phi < 1$. The dictionaries resulting from this parsing algorithm have cardinalities determined essentially by the entropy of the process generating the data and most of the parsed words contain large nested sub-words from the same dictionaries. In this way, our algorithm achieves an asymptotically optimal compression rate for almost every realization of every stationary ergodic process.

Our approach draws inspiration from various fundamental

observations. The main source of inspiration is of course the classical LZ algorithm for 1-dimensional data arrays. In addition, we should mention the results by Ornstein and Weis [4], which relate the recurrence time and the entropy of stationary ergodic processes and show that, in the LZ algorithm, when the string length n is large enough, most of the string is parsed into words of length roughly $\log(n)/h$, where h is the entropy of the process. An insightful overview on ergodic theory and recurrence theory in the context of data compression has been given by Shields [5]. Some of our derivations are guided by his invaluable book on the ergodic theory of discrete sample paths [6]. Finally, we take advantage of the tools elaborated in our previous work on universally typical sets for multidimensional ergodic sources [7].

It is worth mentioning that the 1-dimensional special case of our algorithm is different from the LZ algorithm (our algorithm allows for word repetitions). However, it is possible to regard the LZ algorithm as a limiting case of a modification of our algorithm. We provide more details on this relationship in Section VI.

This paper is organized as follows. Section II describes our settings and gives basic definitions. In Section III we discuss two types of non-sequential universal codes. The first one encodes the data by a single pointer to a universally typical library, and the second one parses the data regularly into blocks of constant shape and size. The main contribution of this work is contained in the following two sections. In Section IV we present our multidimensional sequential recurrence-based algorithm of Lempel Ziv type (Algorithm 6). In Section V we prove that this algorithm is universally asymptotically optimal (Theorem V.1). Section VI contains a few final remarks.

II. Settings

Consider an alphabet $\mathcal{A}$ of finite cardinality $A = |\mathcal{A}| < \infty$, and consider the non-negative orthant $\mathbb{Z}^d_{\geq 0}$ of the d-dimensional integer lattice $\mathbb{Z}^d$, for some finite d. We denote the set of cubical $n\times\cdots\times n$ arrays with entries from $\mathcal{A}$, called n-words, by $\Sigma^n := \mathcal{A}^{\Lambda_n}$, where

$$\Lambda_n := \{0, 1, \ldots, n-1\}^d \subset \mathbb{Z}^d_{\geq 0}.$$

The set of all cubical arrays is denoted by $\Sigma^* := \cup_{n\in\mathbb{N}}\Sigma^n$ and the set of infinitely extended arrays by $\Sigma := \mathcal{A}^{\mathbb{Z}^d_{\geq 0}}$.

Let $\mathfrak{A}^{\mathbb{Z}^d_{\geq 0}}$ denote the σ-algebra of subsets of Σ generated by cylinder sets of the form

$$[w] := \{x \in \Sigma : x(\mathbf{i}) = w(\mathbf{i}), \mathbf{i} \in \Lambda\}$$

for some $w \in \mathcal{A}^\Lambda$, $\Lambda \subset \mathbb{Z}^d_{\geq 0}$, $|\Lambda| < \infty$. Given a subset C of $\mathcal{A}^\Lambda$, the corresponding cylinder set is denoted $[C] := \cup_{w\in C}[w]$. Let $\mathbb{P}$ denote the set of probability measures over $(\Sigma, \mathfrak{A}^{\mathbb{Z}^d_{\geq 0}})$.

Let $\sigma_{\mathbf{r}}$ denote the natural lattice translation by a vector $\mathbf{r} \in \mathbb{Z}^d_{\geq 0}$ acting on Σ by $\sigma_{\mathbf{r}}x(\mathbf{i}) := x(\mathbf{i}+\mathbf{r})$. We use the same notation $\sigma_{\mathbf{r}}$ to denote the induced action on an element ν of $\mathbb{P}$, $\sigma_{\mathbf{r}}\nu(E) := \nu(\sigma_{\mathbf{r}}^{-1}E)$ for all $E \in \mathfrak{A}^{\mathbb{Z}^d_{\geq 0}}$. Here $\sigma_{\mathbf{r}}^{-1}E := \sigma_{-\mathbf{r}}E$ and $\sigma_{\mathbf{r}}E := \cup_{x\in E}\sigma_{\mathbf{r}}x$. The set of all stationary (translation-invariant) elements of $\mathbb{P}$ is denoted by $\mathbb{P}_{\text{stat}}$, i.e., $\nu \in \mathbb{P}_{\text{stat}}$ if $\sigma_{\mathbf{r}}\nu = \nu$ for each $\mathbf{r} \in \mathbb{Z}^d_{\geq 0}$. The stationary measures $\nu \in \mathbb{P}_{\text{stat}}$ which cannot be decomposed as proper convex combinations $\nu = \lambda_1\nu_1 + \lambda_2\nu_2$ with $\nu_1 \neq \nu \neq \nu_2$ and $\nu_1, \nu_2 \in \mathbb{P}_{\text{stat}}$ are called ergodic. The corresponding subset of $\mathbb{P}_{\text{stat}}$ is denoted $\mathbb{P}_{\text{erg}}$.

We denote by ν^n the restriction of the measure ν to the cubical block Λ_n, obtained by the projection $\Pi_n : x \in \Sigma \to x^n \in \Sigma^n$ with $x^n(\mathbf{i}) = x(\mathbf{i})$, $\mathbf{i} \in \Lambda_n$. For an arbitrary finite set $\Lambda \subset \mathbb{Z}^d_{\geq 0}$, the corresponding projection is defined similarly, $\Pi_\Lambda : x \in \Sigma \to x^\Lambda \in \mathcal{A}^\Lambda$.

The entropy rate $h(\nu)$ of a stationary measure $\nu \in \mathbb{P}_{\text{stat}}$ is defined as limit of the scaled n-word entropies:

$$H(\nu^n) := -\sum_{x\in\Sigma^n} \nu^n(w)\log\nu^n(w),$$
$$h(\nu) := \lim_{n\to\infty}\frac{1}{n^d}H(\nu^n).$$

Here and in the following we write $\log$ for the base-A logarithm $\log_A$.

A regular k-block parsing of $x(\Lambda) \in \mathcal{A}^\Lambda$ is the list of sub-arrays $x(\Lambda_k + \mathbf{r} + \mathbf{p})$ for all $\mathbf{r} \in k\cdot\mathbb{Z}^d_{\geq 0}$ with $\Lambda_k + \mathbf{r} + \mathbf{p} \subseteq \Lambda$, for some fixed $\mathbf{p} \in \mathbb{Z}^d_{\geq 0}$. Forgetting about the relative position of the sub-arrays, we identify $\Pi_{\Lambda_k+\mathbf{r}+\mathbf{p}}x \sim \Pi_k\sigma_{\mathbf{r}+\mathbf{p}}x \in \mathcal{A}^{\Lambda_k}$. This yields a list of k-words. We will be mainly interested in parsings where most of Λ is covered by blocks $\Lambda_k + \mathbf{r} + \mathbf{p}$ that are contained in Λ.

The empirical non-overlapping k-block probability distribution of an array x over the sites Λ_n is defined by the relative frequencies of the distinct k-words occurring in the regular k-block parsing of x^n with $\mathbf{p} = 0$,

$$\tilde{\mu}^{k,n}_x(w) := \frac{1}{\lfloor\frac{n}{k}\rfloor^d}\sum_{\mathbf{r}\in\Lambda_{\lfloor\frac{n}{k}\rfloor}} \mathbb{1}_{[w]}(\sigma_{k\cdot\mathbf{r}}x) \quad \forall w \in \Sigma^k.$$

III. Universal Block Codes

In this section we discuss two conceptionally simple but instructive approaches to encode multidimensional data, their drawbacks and possible remedies. This will serve to prepare key concepts behind our sequential algorithm (Section IV) and its optimality proof (Section V).

A. Universally Typical Sets Coding

Universally typical sets define universal codes in a natural way. Given a universally typical set, almost every array can be encoded by its index in that universally typical set. Algorithm 1 is an example based on the universally typical sets that we describe in the following.

Input: data array x^n
Output: compressed array
1: Fix $k = \left\lfloor \sqrt[d]{\log n^d} \right\rfloor$
2: Compute the empirical per-site k-block entropy $h^{k,n}_x := \frac{1}{k^d}H(\tilde{\mu}^{k,n}_x)$
3: Encode x^n by n and its index in $\mathscr{T}_n(h^{k,n}_x)$

Alg. 1. Universally typical set coding algorithm.

Let $\mathscr{T}_n(h_0) \subseteq \Sigma^n$ denote the set of n-words x^n with empirical non-overlapping k-block distributions $\tilde{\mu}^{k,n}_x$ of entropy

$$C_m^{\mu}(\delta) := \{w \in \Sigma^m \colon A^{-m^d(h+\delta)} \leq \mu^m(w) \leq A^{-m^d(h-\delta)}\}. \tag{1}$$

$$\mathcal{T}_k^{\mu}(\delta, m) := \Big\{w \in \Sigma^k : \sum_{\substack{\mathbf{r}\in m\cdot\mathbb{Z}^d:\\ (\Lambda_m+\mathbf{r}+\mathbf{p})\subseteq\Lambda_k}} \mathbb{1}_{[C_m^{\mu}]}(\sigma_{\mathbf{r}+\mathbf{p}}[w]) \geq (1-\delta)\left(\tfrac{k}{m}\right)^d \text{ for some } \mathbf{p} \in \Lambda_m\Big\}. \tag{2}$$

$H(\tilde{\mu}_x^{k,n})$ at most $k^d h_0$, with $k = \lfloor \sqrt[d]{\log n^d} \rfloor$. This set is asymptotically universally typical. More precisely, for each $\mu \in \mathbb{P}_{\text{erg}}$ with $h(\mu) < h_0$, the probability $\mu^n(\mathscr{T}_n(h_0))$ tends to one as n tends to infinity. Furthermore, this set has a log-cardinality of order $n^d h_0$. See [7, Theorem 3.1] for a proof of these statements. Hence x^n can be encoded by a string of length of order $n^d h_0$. We explain the construction more precisely in the following.

Let $h_x^{k,n} := H(\tilde{\mu}_x^{k,n})/k^d$ denote the per-site empirical non-overlapping k-block entropy of an n-word x^n. Let the elements of $\mathscr{T}_n(h_0)$ be indexed in order of increasing $h_x^{k,n}$. This guarantees that the index of an element x^n in $\mathscr{T}_n(h_0)$ is the same for all $h_0 \geq h_x^{k,n}$. Now, for any $\mu \in \mathbb{P}_{\text{erg}}$, the empirical entropy converges to the true entropy, $\lim_{n\to\infty} h_x^{k,n} = h(\mu)$ for μ-almost every x. See [7, Theorem 3.6] for a proof of this statement.

In turn, for almost every realization x of any $\mu \in \mathbb{P}_{\text{erg}}$, as n tends to infinity, for any $\epsilon > 0$, the array x^n can be encoded by its side-length n and its index in $\mathscr{T}_n(h_x^{k,n}) \subseteq \mathscr{T}_n(h(\mu)+\epsilon)$. The typical set does not need to be included in the code, since it can be constructed algorithmically from n, k, and $h_x^{k,n}$. Moreover, we fix $k = \lfloor \sqrt[d]{\log n^d} \rfloor$ and note that $h_x^{k,n}$ can be recovered from n and the index of x^n. This results in an optimal code with length of order $n^d h(\mu)$.

A problem of Algorithm 1 is that it requires the entire data array to be fed at once. If at a given time, after having compressed the array x^n, more data needs to be compressed, then the block size k, the typical set, and the entire code have to be recomputed. Furthermore, although the typical set does not need to be included in the code, constructing it requires exponential running time in n^d, and storing it locally for all k, n, and h would require enormous resources. These disadvantages disqualify Algorithm 1 for practical purposes.

B. Typical Sampling Sets Coding

Instead of encoding the entire data array x^n by a single index, a natural approach is to divide x^n into k-blocks, and to encode each block by its index in a typical sampling set of k-words. Demanding that k be at most of log-order n, the typical sampling set has an arbitrarily small size, relative to n, and can be included in the code. Algorithm 2 parses x^n regularly into non-overlapping k-blocks, and encodes each block by an index to its first occurrence in the parsing. In the following we sketch a proof of the optimality of this algorithm.

Given a measure $\mu \in \mathbb{P}_{\text{stat}}$ with entropy rate $h = h(\mu)$ and a positive real number $\delta > 0$, the level-δ entropy typical set of m-words $C_m^{\mu}(\delta)$ is defined in (1).

For any $k \geq m$, the k-word typical sampling set $\mathcal{T}_k^{\mu}(\delta, m)$ is defined in (2). This is the set of all k-words which have a regular m-block parsing with at least a $(1-\delta)$-fraction of the resulting words contained in the level-δ entropy typical set. As shown in [7, Theorem 3.5] (see Theorem V.4), this set has log-cardinality of order $k^d h(\mu)$. Furthermore, for almost every x, most k-words in the regular k-block parsing of x^n belong to the typical sampling set $\mathcal{T}_k^{\mu}(\delta, m)$. More precisely, for appropriately chosen m, one has $\tilde{\mu}_x^{k,n}(\mathcal{T}_k^{\mu}(\delta, m)) > 1-\alpha$ for any $\alpha \in (0, 1/2)$, when k and n are large enough. Hence most of the k-blocks of x^n are encoded by pointers to a set of log-cardinality of order $k^d h(\mu)$. This implies the asymptotic optimality of Algorithm 2, in the limit where both k and n tend to infinity, with n sufficiently larger than k.

Input: data array x^n
Output: compressed array
1: Fix $k = \left\lfloor \sqrt[d]{\log n^d} \right\rfloor$
2: Parse x^n regularly into k-blocks
3: Encode the list of distinct k-words
4: **for all** k-blocks **do**
5: Encode word by its index in the k-word library
6: **end for**

Alg. 2. Typical sampling set coding algorithm.

In Algorithm 2, one has to include the set of sampled k-words in the code. In contrast to the universally typical sets used in Algorithm 1, this set cannot be reconstructed from a few parameters. However, choosing $k^d \sim \log n^d$, this set is arbitrarily small, and including it in the code does not affect the compression ratio. It is worth mentioning that, if the entropy rate $h(\mu)$ of the process realizing the data is known, then a tight upper bound for the block size of an optimal code is $k = \left\lfloor \sqrt[d]{\frac{\log n^d}{h+\epsilon}} \right\rfloor$, $\epsilon > 0$.

Algorithm 2 remedies the problem of dealing with the enormous typical libraries affecting Algorithm 1. On the other hand, it still suffers from the non-sequentiality problem. Asymptotic optimality is only guaranteed if the block size k increases with the array size n, meaning that every time more data needs to be compressed, the entire code has to be recomputed.

A remedy to this problem is to choose k adaptively, depending on n, in such a way that only the new data entries are parsed with the updated block size. This is precisely the idea of the sequential algorithm that we present in the next section.

IV. Sequential Algorithm

This section contains the description of our sequential recurrence based parsing and coding algorithm for multidi-

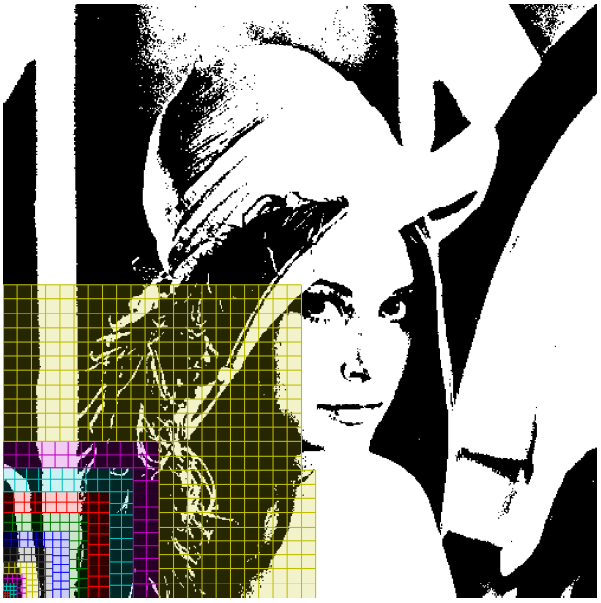

Fig. 3. Illustration of a sequential recurrence-based parsing applied to a black-and-white 512×512 pixel image, with block size increasing after having parsed a cubical region when the proportion of distinct words with repetitions surpasses a certain threshold.

mensional data. The algorithm has two components: the first component constructs parsings of multidimensional data based on a word recurrence criterion. The second component is a coding scheme based on pointers to sub-words of a given parsing. We start with the definition of the coding scheme, in IV-A. Then we define the parsing scheme, in IV-B. An illustration of the parsings generated by our algorithm is given in Figure 3. In Section V we will show that the parsing algorithm generates parsings for which the coding algorithm works optimally.

A. The Coding Algorithm

A parsing P_m of an array $x^m = x(\Lambda_m)$ is a list of blocks $\lambda_1, \ldots, \lambda_N \subseteq \Lambda_m$ with $\cup_{i=1}^N \lambda_i = \Lambda_m$, together with the list of words $w_1 = x(\lambda_1), \ldots, w_N = x(\lambda_N)$. Clearly, the array x^m is fully described by the words $w_1, \ldots, w_N$ and their relative positions in Λ_m.

Given a word $w_l = x(\lambda_l)$, we will consider the set of sub-words of w_l, defined by $W_l := \{x(\lambda') \colon \lambda' \subseteq \lambda_l\}$. Here each $x(\lambda')$ is understood as an element of $\mathcal{A}^{\lambda' - \min \lambda'}$, where $\min \lambda'$ denotes the smallest site of λ' according to the lexicographic order.[1] The sub-words $x(\lambda') \in W_l$ with $\min \lambda' = \min \lambda_l$ are called leading sub-words. The smallest site of the block of a leading sub-word is aligned with the smallest site of the block of the containing word.

Having a parsing of x^m at hand, i.e., a list of blocks and words $P_m = (\lambda_1, \ldots, \lambda_N, w_1, \ldots, w_N)$, we encode x^m by a sequence $C(x^m) = (C_1, \ldots, C_N)$, where $C_i = (p_i, s_i, b_i)$ is the code of w_i and its relative position in Λ_m. The code C_i consists of the following:

[1]The lexicographic order of $\mathbb{Z}^d$ is defined by $(\mathbf{r}_1, \ldots, \mathbf{r}_d) < (\mathbf{s}_1, \ldots, \mathbf{s}_d)$ iff $\exists j \in \{1, \ldots, d\}\colon\ \mathbf{r}_i = \mathbf{s}_i\ \forall i \in \{1, \ldots, j-1\}$ and $\mathbf{r}_j < \mathbf{s}_j$.

1) A pointer p_i, which specifies the smallest $j \in \{1, \ldots, i-1\}$ for which $w_i(\lambda + \mathbf{r}) = w_j(\lambda + \mathbf{s})$ for λ having the largest possible cardinality. Here $\lambda + \mathbf{r} \subseteq \lambda_i$ and $\lambda + \mathbf{s} \subseteq \lambda_j$. In case of ambiguity, we choose the lexicographic smallest $\mathbf{r}, \mathbf{s} \in \mathbb{Z}^d_{\geq 0}$. If the word w_i does not contain any sub-word of any previous word, we set $p_i = \emptyset$. In other words, we consider the largest element in W_i that is contained in some W_j with $j < i$ and set p_i equal to the smallest j for which W_j contains this element.
2) A descriptor s_i of λ_i, λ, $\mathbf{r}$, $\mathbf{s}$. That is, a descriptor of the shape of λ_i, its relative position in Λ_m, the shape of λ, and its relative position in λ_i and in λ_{p_i}.
3) A sequence b_i of the symbols contained in w_i minus the sub-word $w_i(\lambda + \mathbf{r})$, listed according to the lexicographic site order.

The code $C(x^m)$ is determined uniquely by the parsing P_m and gives a full description of the array x^m.

In the following we will consider parsings where all blocks λ_i are cubical; that is, where each λ_i is a translate of some Λ_k. Furthermore, we will restrict the set of possible pointers and consider only pointers from leading cubical sub-words to leading cubical sub-words; that is, with $\lambda = \Lambda_l$ for some $l \leq k$, and $\mathbf{r} = \mathbf{s} = 0$. The pointer structures that we have in mind are illustrated in Figure 4.

As already mentioned in the introduction, the quality of the compression algorithm depends crucially on the properties of the parsing. A good parsing should consist mainly of words that contain large sub-words of previous words. Furthermore, the amount of block overlaps should be as small as possible. The parsing algorithm that we describe next produces parsings with the desired properties.

B. The Parsing Algorithm

Our parsing algorithm has a parameter $0 < \phi < 1$. This parameter will play the role of a threshold for the amount of parsed-word recurrences triggering an increase of the parsing block size.

Given a parsing P_m of x^m, we denote $P_{m,k}$ the list of k-blocks in P_m, and $P'_{m,k}$ the list of k-blocks holding k-words that occur at least twice in P_m. We consider the following function that quantifies the amount of k-word recurrences in P_m:

$$J(P_m, k) := |P'_{m,k}|/|P_{m,k}|. \tag{3}$$

When the parsing P_m does not contain any k-words, i.e., $P_{m,k} = \emptyset$, we set $J(P_m, k) = 0$.

Our parsing algorithm proceeds as follows:

0) Fix a recurrence threshold $0 < \phi < 1$.
1) The input of each iteration is a piece of data $x(\Lambda_n \setminus \Lambda_m)$, $m < n$, together with the parsing P_m of x^m. In the first iteration, $m = 0$ and $P_m = \emptyset$.
2) The parsing block size is set equal to the smallest $k \in \mathbb{N}$ for which the amount of k-word recurrences in the current parsing P_m is below the threshold ϕ; that is, the smallest k for which $J(P_m, k) < \phi$.
3) Define the region to be parsed in this iteration. For the current m and k, this is the k-boundary of Λ_m, defined

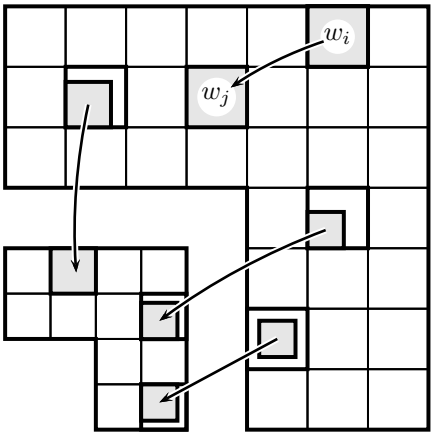

Fig. 4. Illustration of the pointers described in IV-A. Each white square represents a block in a parsing of a two-dimensional array. The blocks are enumerated according to the order in which they were parsed, or just according to the lexicographic order of their smallest sites. The shaded regions with outbound arrows represent largest sub-blocks of parsed blocks, which hold sub-words of previously parsed words. The first occurrences of these sub-words (as sub-words of other parsed words) are shown as shaded regions with corresponding inbound arrows. The last arrow from top to bottom shows an example with non-leading sub-words.

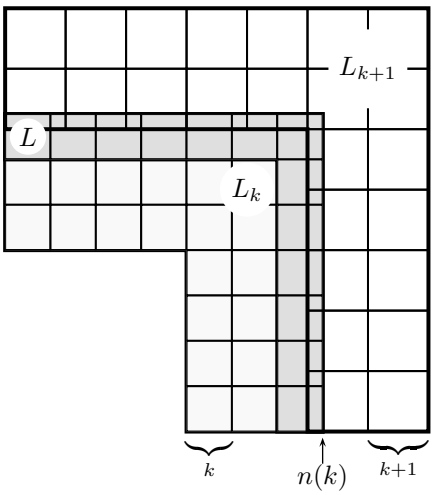

Fig. 5. Illustration of the parsings described in IV-B. The region L_k (shown in gray) represents the set of sites of a two-dimensional array that are parsed with k-blocks. Within this region, the parsing is regular and has no block overlaps. After having parsed the last L boundary region of L_k (shown in dark gray), the amount of k-word recurrences surpasses the specified threshold ϕ, i.e., $J(P_{n(k),k}) > \phi$, and the parsing block size is increased to $k+1$. The first L region of L_{k+1} may overlap with the last L region of L_k. This happens when $n(k) \bmod (k+1) \neq 0$.

as $L := \Lambda_{\lfloor \frac{m}{k} \rfloor k+k} \setminus \Lambda_{\lfloor \frac{m}{k} \rfloor k}$. This definition allows a small overlap of L and Λ_m at the iterations where the parsing block size k has changed.

4) The array region $x(L)$ is parsed regularly into k-blocks. The blocks in the parsing of L are enumerated according to the lexicographic order of their smallest sites.
5) At this point, the data that still needs to be parsed is $x(\Lambda_n \setminus (\Lambda_m \cup L))$. Set $m \leftarrow \lfloor m/k \rfloor k + k$.
6) Repeat the steps 1–5 until Λ_n is exhausted. At the last iteration it may happen that $\lfloor m/k \rfloor k + k > n$. In such a case, set the region L as $\Lambda_n \setminus \Lambda_{n-k}$, or, alternatively, set $k = n - m$ and $L = \Lambda_n \setminus \Lambda_m$.

The parsing structure generated by this algorithm is illustrated in Figure 5. The parsing algorithm, together with the corresponding coding described above, is summarized in Algorithm 6. We denote this compression algorithm SRU_ϕ.

Input: data $x(\Lambda_n \setminus \Lambda_m)$ and parsing P_m of x^m with corresponding code $(C_1, \ldots, C_N)$
Output: extension of P_m to a parsing P_n of x^n with corresponding extended code $(C_1, \ldots, C_N, C_{N+1}, \ldots, C_{N'})$

$k \leftarrow 1$
while $m \leq n - k$ **do**
  if $J(P_m, k) < \phi$ **then**
    k-parse the region $L = \Lambda_{\lfloor \frac{m}{k} \rfloor k+k} \setminus \Lambda_{\lfloor \frac{m}{k} \rfloor k+1}$
    for each k-block λ_i in L **do**
      encode λ_i and $w_i = x(\lambda_i)$ by $C_i = (p_i, s_i, b_i)$
    end for
    $m \leftarrow \lfloor \frac{m}{k} \rfloor k + k$
  else
    $k \leftarrow k + 1$
  end if
end while

Alg. 6. Sequential recurrence-based multidimensional parsing and coding algorithm SRU_ϕ. The input declaration illustrates the sequential nature of the algorithm. At an initial stage the input may be x^n, with $m = 0$ and $P_m = \emptyset$.

V. Optimality

Given an array x^n and some fixed ϕ, let P_n denote the parsing of x^n and $C_\phi(x^n)$ the corresponding code of x^n resulting from Algorithm 6. The code $C_\phi(x^n)$ is a string of length $|C_\phi(x^n)|$, with entries from the alphabet $\mathcal{A}$. Clearly, this $\mathcal{A}$-string can be converted to a binary string of length $|C_\phi(x^n)| \lceil \log_2 A \rceil$.

Theorem V.1. *Let $\mu \in \mathbb{P}_{\mathrm{erg}}$ be an ergodic process with entropy rate h_μ. Then, for any $0 < \phi < 1$,*

$$\lim_{n\to\infty} \frac{|C_\phi(x^n)|}{n^d} = h_\mu \quad \textit{for } \mu\textit{-almost every } x.$$

This theorem shows the universal asymptotic optimality of our compression algorithm (Algorithm 6).

A. Outline of the Proof

The proof builds on several propositions and lemmas. Lemma V.2 estimates the typical number of distinct k-words that appear in the parsing. Proposition V.3 estimates the relative volume of the portion of x^n that is parsed into words of nearly largest side-length. Theorem V.5 is a slight generalization of a result from [7]. It estimates the number of distinct k-words that occur in any non-overlapping parsing of a typical realization of an ergodic process. Lemma V.7 makes a statement about the recurrences of sub-words in the parsings generated by Algorithm 6. It shows that most parsed words contain relatively large leading sub-words which are leading sub-words of previously parsed words. With all these tools at hand, we estimate the length of the code, which concludes the proof of Theorem V.1.

B. Formal Proof

In the following, x denotes a realization of an ergodic process $\mu \in \mathbb{P}_{\text{erg}}$ with entropy rate h_μ. We consider the parsing of x generated by Algorithm 6 for the recurrence function J defined in (3) for some fixed threshold $0 < \phi < 1$.

We denote by $n(k)$ the side-length of the parsed region right after the block-size is increased from k to $k+1$, before the first $(k+1)$-blocks are parsed. See Figure 5.

Let $L_k = L_k(x) \subseteq \Lambda_{n(k)}$ denote the union of all L boundary regions with parsing block-size k.

Let $M_k = M_k(x)$ denote the number of distinct k-words parsed in L_k. Let $\tilde{M}_k = \tilde{M}_k(x)$ denote the total number of k-blocks in the parsing of L_k.

The block-size increase condition $J(P_m, k) \geq \phi$ implies that a ϕ-fraction of all k-blocks in the parsing of L_k hold k-words with multiplicity two or more. Hence $(1-\phi)\tilde{M}_k \geq M_k^{(1)}$, where $M_k^{(1)}$ is the number of k-words that occur only once in the parsing of L_k. If L_k is relatively large compared with any of the L boundary regions that it contains, then $M_k^{(1)} \approx (1-\phi)\tilde{M}_k$, and, approximately, $(1-\frac{1}{2}\phi)^{-1} M_k \leq \tilde{M}_k \leq (1-\phi)^{-1} M_k$.

The following lemma gives a typical bound for M_k.

Lemma V.2. *For any $\alpha_0 > 0$, let $B_{\alpha_0}^+ := \{k : M_k \geq A^{k^d(h_\mu+\alpha_0)}\}$, $B_{\alpha_0}^- := \{k : M_k \leq A^{k^d(h_\mu-\alpha_0)}\}$, and $B_{\alpha_0} := B_{\alpha_0}^+ \cup B_{\alpha_0}^-$. Then*

$$\overline{\mathsf{den}}\Big(\bigcup_{k\in B_{\alpha_0}} L_k(x)\Big) = 0 \quad \textit{for } \mu\textit{-almost every } x,$$

where $\overline{\mathsf{den}}(S) := \limsup_{n\to\infty} \frac{|S\cap\Lambda_n|}{n^d}$ denotes the upper asymptotic density of a set $S \subset \mathbb{Z}^d_{\geq 0}$.

Proof: The proof is based on two main ideas. First, for each $k \in B_{\alpha_0}^+$, the parsing of $x(L_k)$ contains too many different words, contradicting a statement about the number of words needed to construct positive-volumes of realizations of ergodic processes. This part of the proof is independent of the criterion J used to construct the parsing. Second, for $k \in B_{\alpha_0}^-$, the parsed words repeat so quickly that the array $x(L_k)$ has a very small empirical entropy and thus it allows for an encoding that beats the entropy bound. In consequence, these data portions must have a vanishing relative volume.

Part one. Consider first the sets $\bigcup_{k\in B_{\alpha_0}^+} L_k(x)$. Assume, contrary to the claim, that there exist a $B > 0$ and an integer sequence (k_i) with $k_i \to \infty$ satisfying

$$\frac{\bigcup_{k\in B_{\alpha_0}^+, k\leq k_i} L_k(x)}{|\Lambda_{n(k_i)}|} \geq B \quad \text{for all } i \in \mathbb{N}.$$

For some $0 < B' \leq B$, let (k_i') be a sequence with $k_i' \to \infty$ and $\frac{k_i'}{k_i} \leq 1$, satisfying

$$\frac{\bigcup_{k\in B_{\alpha_0}^+, k_i'\leq k\leq k_i} L_k(x)}{|\Lambda_{n(k_i)}|} \geq B' \quad \text{for all } i \in \mathbb{N}.$$

Let (k_i^*) be a sequence with $k_i^* \to \infty$ and $\frac{k_i^*}{k_i'} \to 0$. Consider a regular k_i^*-block parsing of $\bigcup_{k\in B_{\alpha_0}^+, k_i'\leq k\leq k_i} L_k(x)$. For each $k \in B_{\alpha_0}^+$, $k_i' \leq k \leq k_i$, the region L_k has at least $M_k \geq A^{k^d(h+\alpha_0)}$ different k-words, and hence the number E of different k_i^*-words in the parsing satisfies

$$E^{\left\lfloor \frac{k}{k_i^*} \right\rfloor^d} A^{k_i^{*}2dk^{d-1}} \geq M_k \geq A^{k^d(h+\alpha_0)}.$$

Here, the first term in the left hand side is the number of cubical side-length $\lfloor k/k_i^* \rfloor^d$ arrays with entries given by k_i^*-words from a set of cardinality E. The second term in the left hand side is an upper bound on the number of ways in which the region $\Lambda_k \setminus \Lambda_{\lfloor k/k_i^* \rfloor k_i^*}$ can be filled with entries from $\mathcal{A}$. In turn,

$$\log E \geq (k_i^*)^d (h + \alpha_0/2).$$

This shows that a regular k_i^*-block parsing of a B'-portion of $\Lambda_{n(k_i)}$ contains at least $A^{(k_i^*)^d(h+\alpha_0/2)}$ different k_i^*-words.

On the other hand, by [7, Theorem 3.5] (see Theorem V.4 below), for any $\alpha > 0$, there is a set $\mathcal{T}_k = \mathcal{T}_k(\alpha)$ with $\log|\mathcal{T}_k| \leq k^d(h+\alpha)$, which suffices to uniformly construct at least a $(1-\alpha)$-portion of x^n, when k and n/k are large enough, for μ-almost every x. Choosing $\alpha < \frac{\alpha_0}{2}$, $\alpha < B'$, as well as k_i^* large enough, this is a contradiction.

Part two. Assume now that there is a $B > 0$ and a sequence $(k_i)_i$ with $k_i \to \infty$, satisfying

$$\frac{\bigcup_{k\in B_{\alpha_0}^-, k\leq k_i} L_k(x)}{|\Lambda_{n(k_i)}|} \geq B \quad \text{for all } i \in \mathbb{N}.$$

For each k, let $G_k = G_k(x)$ denote the union of all k-blocks in the parsing of $L_k(x)$ that hold k-words occurring for the first time in the parsing. Furthermore, let $\bar{G}_k = L_k \setminus G_k$. By the construction of the parsing, $|\bar{G}_k| \geq \frac{\phi}{2}|G_k|$. Choose a sequence $(l_i)_i$ with $l_i \to \infty$ and $\frac{l_i}{k_i} \to 0$. Let $\bar{G}^i = \bigcup_{k\in B_{\alpha_0}^-, k\leq k_i} \bar{G}_k(x)$. Let $\gamma_i := |\bar{G}^i|/|\Lambda_{n(k_i)}| \geq B\frac{\phi}{2}$. We can encode the region $G^i = \Lambda_{n(k_i)} \setminus \bar{G}^i$ of the array x with an optimal l_i-block coding algorithm, which results in a code of length at most $(1-\gamma_i)(n(k_i))^d(h+\tilde{\alpha})$, whereby $\tilde{\alpha}$ can be chosen arbitrarily close to zero for $n(k_i)$ large enough.

The region $\bar{G}^i$ consists of repetitions of words that occur in the region G^i. Hence, by using pointers to the words in G^i, the region $\bar{G}^i$ can be encoded by a string of length at most $\sum_{k\in B_{\alpha_0}^-, k\leq k_i} \log(M_k)\frac{\bar{G}_k}{k^d}$. Here $\bar{G}_k/k^d$ is the number of pointers, $\log(M_k)$ is the length of each pointer, and we omitted lower order terms. By the assumption on the cardinality of M_k for $k \in B_{\alpha_0}^-$, we obtain the upper bound $\sum_{k\in B_{\alpha_0}^-, k\leq k_i} k^d(h-\alpha_0)\bar{G}_k/k^d = \sum_{k\in B_{\alpha_0}^-, k\leq k_i}(h-\alpha_0)\bar{G}_k$.

In total, we can produce a code of $x(\Lambda_{n(k_i)})$ of length $(1-\gamma_i)(n(k_i))^d(h+\tilde{\alpha}) + \gamma_i(n(k_i))^d(h-\alpha_0) \leq (n(k_i))^d(h-\alpha_0')$, for some $\alpha_0' > 0$. This is a contradiction to the non-existence of codes beating the entropy bound infinitely often. ∎

The next proposition shows that, asymptotically, most of Λ_n is parsed in words of size close to the largest k that is typical in the sense of Lemma V.2.

Consider the sequence $K = (k_i^*)$ consisting of all $k \in \mathbb{N}$ for which L_k has the typical number of distinct parsed words; that is, $A^{h_\mu k(1-\alpha_0)} \leq M_k \leq A^{h_\mu k(1+\alpha_0)}$ for some $\alpha_0 > 0$.

For any two integers $l \leq u$, let $|L_l^u| = \sum_{l\leq k\leq u} |L_k|$ denote the volume of all L_k regions that are parsed with blocks of side-length between l and u, counting overlaps.

Proposition V.3. *For any fixed $\alpha > 0$, the quotient $|L_1^{(1-\alpha)k_i^*}|/|L_{(1-\alpha)k_i^*}^{k_i^*}|$ tends to zero as i tends to infinity.*

Proof: Fix some $\alpha_0 < \alpha$. By Lemma V.2, for all i,

$$\begin{aligned}\sum_{k\le(1-\alpha)k_i^*}|L_k| &= \sum_{\substack{k\notin K:\\ k\le(1-\alpha)k_i^*}}|L_k| + \sum_{\substack{k\in K:\\ k\le(1-\alpha)k_i^*}}|L_k| \\ &\le (1+o(1))\sum_{\substack{k\in K:\\ k\le(1-\alpha)k_i^*}} A^{k^d(h_\mu+\alpha_0)} \\ &\le (1+o(1))CA^{(1-\alpha)^d(k_i^*)^d(h_\mu+\alpha_0)+1} \\ &\le C'A^{(1-\alpha)^d(k_i^*)^d(h_\mu+2\alpha_0)}, \end{aligned} \tag{4}$$

for some constants C and C'. On the other hand,

$$\sum_{(1-\alpha)k_i^*\le k\le k_i^*}|L_k| \ge A^{(k_i^*)^d(h_\mu-\alpha_0)}. \tag{5}$$

Choosing α_0 sufficiently small, e.g., satisfying $\alpha_0 < \alpha h_\mu/3$, we have that $(h_\mu-\alpha_0) > (h_\mu+2\alpha_0)(1-\alpha)^d$, and the right hand side of (5) is exponentially larger than the right hand side of (4). ∎

Before proceeding, we recall the following.

Theorem V.4 (Theorem 3.5a in [7])**.** *Let $\mu\in\mathbb{P}_{\text{erg}}$ with entropy rate $h(\mu)=h$ and let $\delta\in(0,\frac{1}{2})$. Then, for all k larger than some $k_0=k_0(\delta)$ there is an $n_0=n_0(k,\delta)$ such that, if $n\ge n_0$, there is a set $\mathcal{T}_k\subseteq\Sigma^k$ of log-cardinality*

$$\frac{\log|\mathcal{T}_k|}{k^d}\le h+\delta,$$

with $\tilde{\mu}_x^{k,n}(\mathcal{T}_k) > 1-\delta$ for μ-almost every x. An example of $\mathcal{T}_k$ are the typical sampling sets $\mathcal{T}_k^\mu(\delta/2,m)$ from (2) with $m\xrightarrow{k\to\infty}\infty$ and $m=o(k)$.

The next theorem is a slight generalization of Theorem V.4. It states that, asymptotically, most words in any non-overlapping k-block parsing of any positive portion of x^n (instead of a regular parsing of all x^n) belong to a set of log-cardinality close to $h(\mu)k^d$, almost surely.

Theorem V.5. *Let $\mu\in\mathbb{P}_{\text{erg}}$ with entropy rate $h(\mu)=h$ and let $\delta>0$, $\delta'>0$. Consider a collection of non-overlapping k-blocks $\lambda_1,\dots,\lambda_I$ covering a region $\Xi\subseteq\Lambda_n$ with $|\Xi|/|\Lambda_n|\ge\beta>0$. Then, for all k larger than some $k_0=k_0(\delta,\delta')$ there is an $n_0=n_0(k,\delta,\delta')$ such that, if $n\ge n_0$, there is a set $\mathcal{T}_k\subseteq\Sigma^k$ of log-cardinality*

$$\frac{\log|\mathcal{T}_k|}{k^d}\le h+\delta,$$

with $w_i=x(\lambda_i)\in\mathcal{T}_k$ for at least a $(1-\delta')$-fraction of all $i\in I$, for μ-almost every x. Here, the set $\mathcal{T}_k$ does not depend on the specific x.

Proof: The proof follows the steps of the proof of [7, Theorem 3.5]. We show that $\mathcal{T}_k$ can be chosen as the typical sampling sets $\mathcal{T}_k^\mu(\delta/2,m)$ from (2) with $m\xrightarrow{k\to\infty}\infty$ and $m=o(k)$.

Fix some δ, δ', and β. The Shannon-McMillan-Breiman theorem for amenable groups by Ornstein and Weiss [8] states that[2]

$$\lim_{m\to\infty}-\frac{1}{m^d}\log\mu^m(x^m)=h(\mu),\quad \mu\text{-almost surely.}$$

Thus, by the definition (1) of the entropy typical sets $C_m^\mu(\delta)$, there exists an $m_0=m_0(\delta,\delta',\beta,x)$ such that $\mu^m(C_m^\mu)\ge 1-\delta'\delta\beta/8$ for all $m\ge m_0$. Fix such an $m\ge m_0$. The individual ergodic theorem [10] guarantees that the following limit exists for μ-almost every x:

$$\lim_{n\to\infty}\frac{1}{n^d}\sum_{r\in\Lambda_n}\mathbb{1}_{[C_m^\mu]}(\sigma_r x)=\int\mathbb{1}_{[C_m^\mu]}(x)d\mu(x)=\mu^m(C_m^\mu).$$

Therefore, the inequality

$$\begin{aligned}\sum_{r\in\Lambda_{n-m+1}}\mathbb{1}_{[C_m^\mu]}(\sigma_r x) &\ge (1-\delta'\delta\beta/7)(n-m+1)^d \\ &> (1-\delta'\delta\beta/6)n^d \end{aligned}\tag{6}$$

holds eventually almost surely, i.e., for μ-almost every x and $n\ge n_0(x,m)$.

Consider an x and an $n\in\mathbb{N}$ for which (6) is satisfied. Choose a k with $m<k<n$, and consider the non-overlapping k-blocks $\lambda_1,\dots,\lambda_I$ that cover an β-fraction of Λ_n. If k/m and n/k are large enough, then at least $(1-\delta')I$ of the words $w_1,\dots,w_I$ satisfy

$$\frac{1}{(k-m+1)^d}\sum_{\mathbf{s}\in\Lambda_{k-m+1}}\mathbb{1}_{[C_m^\mu]}(\sigma_{\mathbf{s}}[w_i])\ge(1-\delta/4),\tag{7}$$

where $[w_i]:=\{x\in\Sigma\colon x(\mathbf{i})=w_i(\mathbf{i}),\mathbf{i}\in\Lambda_k\}$.

To see this, note that if more than $\delta' I$ of the w_i had more than a $\delta/4$-fraction of 'bad' m-blocks (blocks with contents not in C_m^μ), then the total number of 'bad' m-blocks in x^n would be larger than $\delta' I\frac{\delta}{4}(k-m+1)^d$. For n/k and k/m large enough, this is lower bounded by $\delta'\beta\left\lfloor\frac{n}{k}\right\rfloor^d\frac{\delta}{4}(k-m+1)^d\ge\delta'\beta\frac{\delta}{6}n^d$, contradicting (6).

Now, if $k\ge 8dm/\delta$, the k-words that satisfy (7) have a regular m-block parsing with at least a $(1-\delta/2)(k/m)^d$ words in C_m^μ. This is by the following lemma.

Lemma V.6 (Lemma 3.2 in [7])**.** *Let $k\ge 4dm/\delta$. If $C\subseteq\Sigma^m$ and $w\in\Sigma^k$ satisfy $\sum_{\mathbf{r}\in\Lambda_{k-m+1}}\mathbb{1}_{[C]}(\sigma_{\mathbf{r}}[w])\ge(1-\delta/4)(k-m+1)^d$, then, for some $\mathbf{p}\in\Lambda_m$, $\sum_{\substack{\mathbf{r}\in m\cdot\mathbb{Z}^d:\\(\Lambda_m+\mathbf{r}+\mathbf{p})\subseteq\Lambda_k}}\mathbb{1}_{[C]}(\sigma_{\mathbf{r}+\mathbf{p}}[w])\ge(1-\delta)(k/m)^d$.*

Hence each word satisfying (7) is in the typical sampling set $\mathcal{T}_k^\mu(\delta/2,m)$ defined in (2). As discussed above, this applies to at least a $(1-\delta')$-fraction of the words $w_1,\dots,w_I$. By Theorem V.4, the log-cardinality of $\mathcal{T}_k^\mu(\delta/2,m)$ is upper bounded by $k^d(h+\delta)$. ∎

The next lemma addresses the "nestedness" of the parsings generated by Algorithm 6, that is, the amount of sub-words of parsed words, which are sub-words of previously parsed words. Consider the sequence $(k_i^*(\alpha_0))$ of all k for which L_k has the typical number of distinct parsed words, $A^{k^d(h-\alpha_0)}\le M_k\le A^{k^d(h+\alpha_0)}$, for some $\alpha_0>0$.

[2]Here in fact we only need the convergence in probability, shown in [9], ensuring $\mu(C_m^\mu)\xrightarrow{m\to\infty}1$.

$$|C(\tilde{L}_{k'}^{k^*})| \leq (1-\epsilon)\tilde{\tilde{M}}_{k'}^{k^*}\Big(\underbrace{\log(\sum_{k=1}^{k^*}\tilde{M}_k)}_{1)}+\underbrace{\log(k^*)}_{2)}+\underbrace{(k^*-k')^{d-1}d}_{3)}\Big)+\epsilon\tilde{\tilde{M}}_{k'}^{k^*}\Big(\underbrace{\log(\sum_{k=1}^{k^*}\tilde{M}_k)}_{1)}+\underbrace{\log(k^*)}_{2)}+\underbrace{k^{*d}}_{3)}\Big). \quad (8)$$

Lemma V.7. *For all $\epsilon, \alpha', \alpha$ with $1/2 > \alpha' > \alpha > 0$, there is an $\alpha_0 < \alpha$ and an $i_0 \in \mathbb{N}$ such that for all $i \geq i_0$, at least a $(1-\epsilon)$-fraction of all words parsed in $L_{k'}^{k_i^*(\alpha_0)}$, $k' = \lfloor(1-\alpha)k_i^*(\alpha_0)\rfloor$, have a leading sub-word of side-length larger or equal to $k'' = \lfloor(1-\alpha')k_i^*(\alpha_0)\rfloor$ that is a leading sub-word of a previously parsed word.*

Proof: Assuming that an ϵ-fraction of all words parsed in $L_{k'}^{k^*}$ do not have leading sub-words of side-length larger or equal to k'' that are leading sub-words of previously parsed words, we show that there are too many distinct non-overlapping sub-words covering a positive fraction of $\Lambda_{n(k^*)}$, contradicting Theorem V.5.

We abbreviate $k_i^*(\alpha_0)$ by k^*. The set of leading k''-sub-blocks of the blocks parsed in $L_{k'}^{k^*}$ has a total volume at least $(1-\alpha')|L_{k'}^{k^*}|$. According to Proposition V.3, $|L_1^{k'-1}| \leq \tilde{\epsilon}|L_{k'}^{k^*}|$, where $\tilde{\epsilon}$ becomes arbitrarily small when k^* is large enough. Hence $(1-\alpha')|L_{k'}^{k^*}| \geq ((1-\alpha')-\tilde{\epsilon})|L_1^{k^*}|$. In fact, the parsing contains a set of non-overlapping leading k''-sub-blocks that covers at least a $((1-\alpha')-\tilde{\epsilon})$-fraction of the region $\Lambda_{n(k^*)}$.

By Lemma V.2, the number $\tilde{M}_{k'}^{k^*}$ of words parsed in $L_{k'}^{k^*}$ is lower bounded by $A^{k'^d(h-\alpha_0)}$. Assume that an ϵ-fraction of these words do not have a leading k''-sub-word that is a leading sub-word of any previously parsed word. Each of these words must have a distinct leading k''-sub-word, since otherwise some of them would be sub-words of previously parsed words. The situation is illustrated in Figure 7. Under this assumption, the number of distinct leading k''-sub-words is at least $\epsilon\tilde{M}_{k'}^{k^*}$. Keep in mind that $\tilde{M}_{k'}^{k^*} \geq A^{k'^d(h-\alpha_0)}$.

On the other hand, choosing k^* large enough, $\tilde{\epsilon}$ becomes arbitrarily small, and $((1-\alpha')-\tilde{\epsilon}) > 0$. Recall that the latter lower bounds the fraction of $\Lambda_{n(k^*)}$ covered by leading k''-sub-blocks in $L_{k'}^{k^*}$. By Theorem V.5, using $\beta = \epsilon((1-\alpha')-\tilde{\epsilon})$, there is a set $\mathcal{T}_{k''}$ of log-cardinality $\log|\mathcal{T}_{k''}| \leq k''^d(h+\delta)$, which contains at least a $(1-\delta)$-fraction of all leading k''-sub-words in $L_{k'}^{k^*}$.

Hence $A^{k''^d(h+\delta)} \geq (\epsilon-\delta)A^{k'^d(h-\alpha_0)}$. This is a contradiction when $\delta < \epsilon$ and $\frac{(1-\alpha')^d}{(1-\alpha)^d}\frac{(h+\delta)}{(h-\alpha_0)} < 1$, which is always the case for sufficiently small δ and α_0. ∎

We have now developed all the tools that we need to complete the proof of the main theorem.

Proof of Theorem V.1: Consider a word w parsed at a given time step. Let $\tilde{w}$ be the largest leading sub-word of w that is the leading sub-word of a previously parsed word. The code of w consists of the following:

1) A pointer to the first previous word w' which contains $\tilde{w}$ as a leading sub-word.
2) A descriptor of the side-length of $\tilde{w}$.
3) An encoding of the boundary entries of w, i.e., the entries of w that are not contained in $\tilde{w}$.

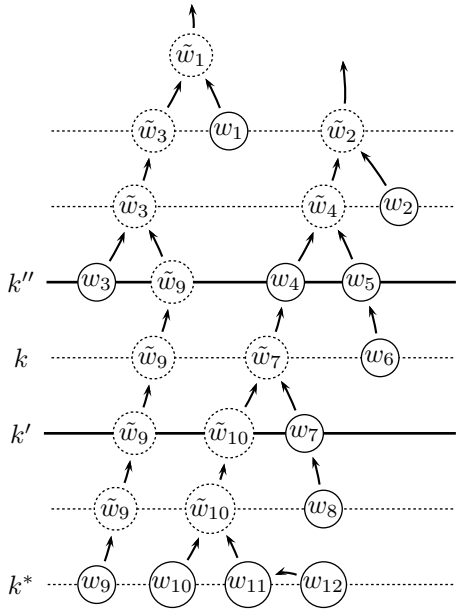

Fig. 7. Sketch of the tree structure of nested words parsed by Algorithm 6. Most words parsed in $L_{k'}^{k^*}$ have leading sub-words that are leading sub-words of words parsed in $L_{k''}^{k^*}$. Assuming the contrary would imply the existence of too many distinct leading k''-sub-words. In the figure, the horizontal lines stand for the side-lengths of the parsed words, w_i represents the i-th parsed word, and a $\tilde{w}_i$ in the horizontal line k represents the leading k-sub-word of w_i in the case that it is not a leading k-sub-word of any previously parsed word w_j, $j < i$. The arrows represent the pointers defined in IV-A item 1. For example, the arrow from $\tilde{w}_{10}$ to $\tilde{w}_7$ signifies that $p_{10} = 7$ and that the considered leading sub-word has side-length k.

Consider the sequence $K = (k_i^*)$ of all k for which L_k has the typical number of blocks, according to Lemma V.2, for some $\alpha_0 > 0$. Let us first assume that at the current time step, the parsing block side-length is equal to some $k^* \in K$. Let $n^* \in \mathbb{N}$ denote the side-length of the smallest cubical region containing the L region being parsed at the current time step. We have $n(k^*-1) \leq n^* \leq n(k^*)$.

Let $\tilde{L}_{k^*}$ denote the set $L_{k^*} \cap \Lambda_{n^*}$. Let $\tilde{\tilde{M}}_{k^*}$ denote the total number of k^*-blocks in $\tilde{L}_{k^*}$. The region $\tilde{L}_{k^*}$ is contained in L_{k^*} and hence $\tilde{\tilde{M}}_{k^*} \leq \tilde{M}_{k^*}$. The number of blocks of side-length from k' to k^* parsed in Λ_{n^*} is given by $\tilde{\tilde{M}}_{k'}^{k^*} := \tilde{M}_{k'}^{(k^*-1)} + \tilde{\tilde{M}}_{k^*}$. By Lemma V.7, a $(1-\epsilon)$-fraction of the words parsed in $\tilde{L}_{k'}^{k^*}$ have large nested sub-words. For the encoding $C(\tilde{L}_{k'}^{k^*})$ of the words parsed in $\tilde{L}_{k'}^{k^*}$, we have the bound (8). By similar arguments as in (4), $\tilde{\tilde{M}}_{k'}^{k^*} \leq \sum_{k=1}^{k^*}\tilde{M}_k \leq cA^{k^{*d}(h+\alpha_0)}$, and hence

$$\begin{aligned}|C(\tilde{L}_{k'}^{k^*})| \leq& (1-\epsilon)\tilde{\tilde{M}}_{k'}^{k^*}\Big(k^{*d}(h+\alpha_0)+o(k^{*d})\Big)\\ &+\epsilon\tilde{\tilde{M}}_{k'}^{k^*}(h+1)k^{*d}\\ \leq& (1+\epsilon')n^{*d}(h+\alpha_0)+\epsilon n^{*d}. \end{aligned} \quad (9)$$

Since ϵ' and α_0 can be chosen arbitrarily small, the right hand side (9) can be given as $n^{*d}h(1+o(1))$. Since $|L_1^{k'-1}|$ is arbitrarily small compared with $|L_{k'}^{k^*}|$ as k^* tends to infinity, we obtain

$$|C(x^{n^*})| \leq n^{*d}h(1+o(1)).$$

Since the relative volume covered by regions L_k with $k \notin K = (k_i^*)$ tends to zero (see Lemma V.2), the code length of these regions is negligible. Hence, for arbitrary n we get the same bound as for n^*,

$$|C(x^n)| \leq n^d h(1+o(1)).$$

This completes the proof. ∎

VI. Remarks

Our proof of the asymptotic optimality of Algorithm 6 also yields the optimality of the modified algorithm that considers arbitrary cubical sub-words of parsed words instead of leading cubical sub-words alone. This additional freedom in the choice of the pointers can only improve the compression ratio.

Our proof shows the asymptotic optimality for any choice of the parameter ϕ. Nevertheless, this parameter may well influence the velocity of convergence of the compression ratio. Fine tuning of our algorithm may be of interest for practical implementations. An extensive empirical study is in preparation.

The 1-dimensional special case of our algorithm differs from the classical LZ algorithm, since our algorithm allows word repetitions and uses a strictly non-decreasing parsing block size. The LZ algorithm can be obtained from our algorithm by setting $\phi = 0$, defining the boundary regions L as $\{m+1,\ldots,k\}$, instead of $\Lambda_{\lfloor m/k\rfloor k+k}\setminus\Lambda_{\lfloor m/k\rfloor k}$, and evaluating the recurrence quantifying function J not on the current parsing but "predictively" on the parsing that one would obtain in the next iteration for a given choice of the parsing block size k for the current iteration. Our proofs do not directly apply to that setting. However, modifications evaluating J predictively and allowing shrinking parsing block size are plausible.

Acknowledgment

We would like to thank the Information Theory of Cognitive Systems Group at the Max Planck Institute MIS for funding and infrastructure for this project.

References

[1] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, 1977.

[2] A. Lempel and J. Ziv, "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. 24, no. 5, 1978.

[3] ——, "Compression of two-dimensional data," *IEEE Transactions on Information Theory*, vol. 32, no. 1, pp. 2–8, 1986.

[4] D. Ornstein and B. Weiss, "Entropy and data compression schemes," *Information Theory, IEEE Transactions on*, vol. 39, no. 1, pp. 78–83, Jan 1993.

[5] P. Shields, "The interactions between ergodic theory and information theory," *Information Theory, IEEE Transactions on*, vol. 44, no. 6, pp. 2079–2093, Oct 1998.

[6] ——, *The Ergodic Theory of Discrete Sample Paths*, ser. Graduate Studies in Mathematics. American Mathematical Society, 1996, vol. 13.

[7] T. Krüger, G. Montúfar, R. Seiler, and R. Siegmund-Schultze, "Universally typical sets for ergodic sources of multidimensional data," *Kybernetika*, vol. 49, no. 6, pp. 868–882, 2013.

[8] D. S. Ornstein and B. Weiss, "The Shannon-McMillan-Breiman theorem for a class of amenable groups," *Isr. J. Math.*, vol. 44, 1983.

[9] J. C. Kieffer, "A generalized Shannon-McMillan theorem for the action of an amenable group on a probability space," *Ann. Probability*, vol. 3, no. 6, pp. 1031–1037, 1975.

[10] E. Lindenstrauss, "Pointwise theorems for amenable groups," *Inventiones Mathematicae*, vol. 146, no. 2, pp. 259–295, 2001.