

# Descriptions of Words over a Partially Commutative Alphabet

Serap A. Savari

Bell Labs, Lucent Technologies, Murray Hill, NJ 07974 USA

## I. INTRODUCTION

The concept of *ordering of events* is fundamental to the study of the dynamic behavior of a system. In a sequential process it is natural to use strings of symbols over some alphabet to specify the temporal ordering of events. The symbols may, for example, correspond to the states, commands, or messages in a computation. A recent prize-winning paper [1] in the area of computing systems applies a lossless data compression algorithm known as SEQUITUR [2] to the the sequence of events or signals determining the control flow or operations of a program's execution. The author of [1] demonstrated that the grammar which is output from SEQUITUR can be exploited to identify performance tuning opportunities via heavily executed subsequences of operations.

The underlying premise in using lossless data compression for this application is the existence of a well-defined linear ordering of events in time. A *partial ordering* of events is a more accurate model for concurrent systems like multiprocessor configurations, distributed systems and communication networks which consist of a collection of distinct processes which communicate with one another or synchronize at times but are also partly autonomous. These complex systems permit independence of some events occurring in the individual processes while others must happen in a predetermined order [3].

Trace theory [4]-[5] is an approach to extending the notions and results pertaining to strings in order to treat the partial ordering of event occurrences in concurrent systems. The idea is to combine the sequence of atomic actions observed by a single witness of a concurrent system with a labeled and undirected *dependence relation* or *noncommutation graph* specifying which actions can be executed independently or concurrently. Two words with symbols over a vertex set  $V$  are *congruent* or *equivalent* with respect to a noncommutation graph  $G$  if each can be obtained from the other through a process of interchanging consecutive letters that are nonadjacent vertices in  $G$ . As an example, if the noncommutation graph  $G$  is given by  $a-b-c-d$ , then the two words  $ddbca$  and  $bdadc$  are congruent since  $ddbca \equiv_G dbdca \equiv_G dbdac \equiv_G dbadc \equiv_G bdadc$ .

There are two special cases of the dependence relation which are standard in information theory. When  $G$  is the complete graph on the vertex set  $V$ , i.e., when there is an edge connecting every pair of vertices, every word over  $V$  is congruent only to itself. At the other extreme, if  $G$  is the empty graph on the vertex set, i.e., no two vertices are adjacent, the equivalence classes on words are called *type classes* [7, p.280] or *composition classes* [8, p.29] in the information theory literature and *rearrangement classes* or *abelian classes* in combinatorics [9, p.184]. Mazurkiewicz [4] dubbed a congruence class of words for an arbitrary noncommutation graph  $G$  a *trace* because they represent traces of processes, i.e., the sequence of states traversed by the process from initialization to termination, in nonsequential systems. Because the word trace has numerous connotations, we will create the terminology *interchange class* to refer to an equivalence class of words.

Motivated by the success of [1] in applying lossless data compression algorithms to a string of events in a sequential system, [6] introduced a compression problem where it is only necessary to reproduce a string which is in the same interchange class as the original string. That paper described some compression schemes for the congruence class of a string that in the best cases can be exponentially more succinct than the optimal grammar-based representation of the corresponding string. This compression problem also appears in the compression of executable code [10]. Executable code or program "binaries" are files whose content must be interpreted by a program or hardware processor which knows exactly how the data inside the file is formatted in order to utilize it. One of the techniques given in [10] for this compression application is "instruction rescheduling," in which instructions can be reordered if the decompressed program is execution-isomorphic to the original. In this paper, we propose a new generalization of Kolmogorov-Chaitin complexity (see, e.g., [11]) called the *interchange complexity* and use a version of the subadditive ergodic theorem [12] to provide sufficient conditions on probabilistic sources so that an extension of the asymptotic equipartition

property [13], [14] to interchange classes holds. We name the average number of bits per symbol needed to represent an interchange class the *interchange entropy*.

## II. THE INTERCHANGE COMPLEXITY

The asymptotic equipartition property is central to the study of lossless data compression. It states that most long sequences from an ergodic source are typical in the sense that their mean self-information per symbol is close to the entropy of the source. A consequence of this result is that the average number of bits per symbol required to losslessly encode the output of an ergodic source is asymptotically bounded from below by the binary entropy of the source. In order to find a counterpart for this lossy compression problem, we begin by considering the least amount of information about an individual string that must be described in order to reproduce another string within the same interchange class. The appropriate framework for this discussion is algorithmic information theory (see, e.g., [15] and [11]). For a finite length string  $x$  over the vertex set  $V$ , we let  $C(x)$  denote the Kolmogorov complexity of  $x$  and refer to [11, §2.1, p. 107] for the basic properties of  $C(x)$ . Let  $V^*$  be the set of all finite words from  $V$  and  $|V|$  denote the cardinality of  $V$ . We define the *interchange complexity* of  $x \in V^*$  with respect to a noncommutation graph  $G$  with vertex set  $V$  by

$$C_i(G, x) = \min\{C(y) \mid y \in V^*, y \equiv_G x\}.$$

$C_i(G, x)$  has the interpretation of being the length of the shortest program for a universal computer that will print out a word  $y$  which is congruent to  $x$  with respect to the noncommutation graph  $G$ .

Suppose that we have words  $u, v, w, x \in V^*$  with  $u \equiv_G w$  and  $v \equiv_G x$ . Then it is easily seen that the words  $uv$  and  $wx$  formed by respectively appending  $v$  to  $u$  and  $x$  to  $w$  satisfies  $uv \equiv_G wx$ . Therefore, the interchange complexity is *almost subadditive*. In particular, to bound  $C_i(G, uv)$  from above, we follow [11, p.102] and observe that one way to produce a string equivalent to  $uv$  is to use a shortest program  $p$  to find a string  $w$  equivalent to  $u$ , a shortest program  $q$  to construct a string  $x$  congruent to  $v$ , a means to schedule the two programs to produce  $w$  followed by  $x$ , and an identification of the programs  $p$  and  $q$ . Using this encoding technique we find that

$$C_i(G, uv) \leq C_i(G, u) + C_i(G, v) + 2 \log_2(\min(l(u), l(v))) + 0(1). \quad (1)$$

Since (1) is a relaxed subadditivity relation satisfying a special constraint, we can use the following result from [16].

*Theorem II.1* (De Bruijn-Erdős (1952)) Suppose  $\phi$  is a positive and nondecreasing function that satisfies

$$\int_1^\infty \frac{\phi(t)}{t^2} < \infty.$$

If  $\{x_n\}$  satisfies the relaxed subadditivity relation

$$x_{n+m} \leq x_n + x_m + \phi(n+m), \quad \frac{n}{2} \leq m \leq 2n$$

then as  $n \rightarrow \infty$ ,  $x_n/n$  converges to  $\gamma = \inf_{m \geq 1} x_m/m$ .

The preceding theorem implies that the asymptotic per symbol information content needed to convey a word equivalent to the original bound is well-defined. More specifically, we have

*Proposition II.2:* For any word  $u_1 \dots u_n$  with  $u_i \in V$ ,  $i \in \{1, 2, \dots, n\}$ , we have that  $n$  approaches infinity,  $n^{-1}C_i(G, u_1 \dots u_n)$  converges to  $\inf_{m \geq 1} m^{-1}C_i(G, u_1 \dots u_m)$ .

We next wish to find a probabilistic version of Proposition II.2. The appropriate frame of reference is subadditive ergodic theory. We apply a theorem from [12] to obtain the following result.

*Theorem II.3* (A.E.P. for interchange classes) Let  $U_1, U_2, \dots$  be the random output of a finite alphabet, discrete, stationary, and ergodic source or of a finite state and finite alphabet unifilar Markov source. Then with probability 1,

$$\lim_{n \rightarrow \infty} \frac{C_i(G, U_1 U_2 \dots U_n)}{n} = \lim_{n \rightarrow \infty} \frac{E[C_i(G, U_1 U_2 \dots U_n)]}{n} = \inf_{m \geq 1} \frac{E[C_i(G, U_1 U_2 \dots U_m)]}{m}.$$

For probabilistic sources  $P$  in which the random variables  $n^{-1}C_l(G, U_1 \dots U_n)$  converge almost surely or in probability to  $\lim_{n \rightarrow \infty} n^{-1}E[C_l(G, U_1 \dots U_n)]$ , we name the latter expression the *interchange entropy* and denote it by  $H_l(G, P)$ . Just as the asymptotic equipartition property for strings leads to a notion of typical sequences which all have about the same probability and together constitute the possible outputs of the source with high probability (see, e.g., [7, Ch. 3]), Theorem II.3 provides a comparable concept of typical interchange classes. Most long strings require close to  $H_l(G, P)$  bits per symbol to describe an equivalent string with respect to the noncommutation graph  $G$ . It follows that the typical sequences of length  $n$  fall into approximately  $2^{nH_l(G, P)}$  typical interchange classes.

It is generally considered to be difficult to determine or even bound the limiting constants obtained by a subadditivity argument (see, e.g., [17, pp.114,116]). In [18], we consider a discrete, memoryless source emitting symbols from the vertex set  $V$  of a noncommutation graph  $G$  with a probability distribution  $P$  on the vertex set. In the case where  $G$  is the complete  $k$ -partite graph  $K_{m_1, m_2, \dots, m_k}$ , we can obtain the exact expression for  $H_l(G, P)$ . We represent the vertex set  $V$  by  $V = V_1 \cup V_2 \cup \dots \cup V_k$ ,  $|V_i| = m_i$ ,  $i \in \{1, 2, \dots, k\}$  and the elements of  $V_i$  are labeled  $v_{i,j}$ ,  $i \in \{1, 2, \dots, k\}$ ,  $j \in \{1, 2, \dots, m_i\}$ . We assume that every pair of vertices  $\{v_{i,j}, v_{l,n}\}$ ,  $v_{i,j} \in V_i$ ,  $v_{l,n} \in V_l$ ,  $l \neq i$ , is an edge and that there is no edge consisting of two vertices from the same subset of vertices  $V_i$  for any  $i \in \{1, 2, \dots, k\}$ . Define  $Q_i = \sum_{j=1}^{m_i} P(v_{i,j})$ ,  $i \in \{1, 2, \dots, k\}$ . Then in [18] we establish the following result.

*Theorem II.4:*

$$H_l(K_{m_1, m_2, \dots, m_k}, P) = H(P) - \sum_{S=2}^{\infty} \log_2(S) \sum_{i: m_i \geq 2} (1 - Q_i) \left( Q_i^S - \sum_{j=1}^{m_i} \left( \frac{P(v_{i,j})}{1 - Q_i + P(v_{i,j})} \right)^S \right).$$

In [18] we establish upper and lower bounds for  $H_l(G, P)$  for general graphs.

#### ACKNOWLEDGMENT

The author would like to thank K. Etesami, R. Alur, S. Guha, M. Yannakakis, and S. Chaudhuri for formulating and suggesting the problem of compressing partial orders, for helpful discussions, and for providing two drafts of [6].

#### REFERENCES

- [1] J. Larus, "Whole program paths," *ACM SIGPLAN Conf. Prog. Lang. Des. Implem.* 259-269, Atlanta, GA, May 1999.
- [2] C. G. Nevill-Manning and I. H. Witten, "Linear-time, incremental hierarchy inference for compression," *Proc. 1997 I.E.E.E. Data Comp. Conf.* 3-11, Snowbird, UT, March 1997.
- [3] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Comm. ACM* 21, 558-565, 1978.
- [4] A. Mazurkiewicz, "Trace theory," in W. Brauer et. al., ed., *Petri Nets, Applications and Relationship to other Models of Concurrency*, Lecture Notes in Computer Science 255, 279-324, Springer, Berlin, 1987.
- [5] V. Diekert and Y. M etivier, "Partial commutation and traces," in G. Rozenberg and A. Salomaa, ed., *Handbook on Formal Languages: Beyond Words*, Volume III, 457-534, Springer, Berlin, 1997.
- [6] R. Alur, S. Chaudhuri, K. Etesami, S. Guha and M. Yannakakis, "Compression of partially ordered strings," submitted for publication, 2003.
- [7] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, New York 1991.
- [8] I. Csisz ar and J. K orner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, Akad emiai Kiad o, Budapest, 1997.
- [9] D. Foata, "Rearrangements of words," in M. Lothaire, ed., *Combinatorics on Words*, 184-212, Cambridge University Press, Cambridge, 1997.
- [10] M. Drini c and D. Kirovski, "PPMexe: PPM for compressing software," *Proc. 1997 I.E.E.E. Data Comp. Conf.* 192-201, Snowbird, UT, March 2002.
- [11] M. Li and P. Vit anyi, *An Introduction to Kolmogorov Complexity and Its Applications*, Second Edition, Springer, New York, 1997.
- [12] Y. Derriennic, "Un th eor eme ergodique presque sous-additif," *Ann. Prob.* 11, 669-677, 1983.
- [13] C. E. Shannon, "A mathematical theory of communication," *Bell System Tech. J.* 27, 379-423, 623-656, 1948.
- [14] B. McMillan, "The basic theorems of information theory," *Ann. Math. Stat.* 24, 196-219, 1953.
- [15] A. N. Kolmogorov, "Logical basis for information theory and probability theory," *I.E.E.E. Trans. Inform. Theory* IT-14, 662-664, 1968.
- [16] N. G. DeBruijn and P. Erd os, "Some linear and some quadratic recursion formulas I," *Indag. Math.* 13, 374-382, 1952.
- [17] W. Szpankowski, *Average Case Analysis of Algorithms on Sequences*, Wiley, New York, 2001.
- [18] S. A. Savari, "Concurrent processes and the interchange entropy," submitted to the 2003 IEEE International Symposium on Information Theory.