Entropy-Compressed Indexes for Multidimensional Pattern Matching

Roberto Grossi
University of Pisa, Italy

Ankur Gupta
Duke University, USA

Jeffrey Scott Vitter
Purdue University, USA

Abstract

In this talk, we will discuss the challenges involved in developing a multidimensional generalizations of compressed text indexing structures. These structures depend on some notion of Burrows-Wheeler transform (BWT) for multiple dimensions, though naive generalizations do not enable multidimensional pattern matching. We study the 2D case to possibly highlight combinatorial properties that do not emerge in the 1D case. We also present related work in 2D pattern matching and indexing.

Introduction

Suffix arrays and suffix trees are ubiquitous data structures at the heart of several text and string algorithms. They are used in a wide variety of applications, including pattern matching, text and information retrieval, Web searching, and sequence analysis in computational biology [14]. Compressed suffix arrays [13, 18, 19] and opportunistic FM-indexes [7, 8] represent new trends in the design of advanced indexes for full-text searching of documents, in that they support the functionalities of suffix arrays and suffix trees, which are more powerful than classical inverted files, yet they also overcome the aforementioned space limitations by exploiting, in a novel way, the notion of text compressibility and the techniques developed for succinct data structures and bounded-universe dictionaries.

Grossi and Vitter [13] developed the compressed suffix array using $2n \log |\Sigma|$ bits in the worst case with $o(m)$ searching time. Sadakane [18, 19] related the space bound to the order-0 empirical entropy H_0. Ferragina and Manzini devised the FM-index [7, 8], which is based on the Burrows-Wheeler transform (BWT) and is the first to encode the index size with respect to the hth-order empirical entropy H_h of the text. Navarro [17] recently developed an index requiring $4nH_h + o(n)$ bits, and boasts fast search. Grossi, Gupta, and Vitter [11] exploited the higher-order entropy H_h of the text to represent a compressed suffix array in just $nH_h + O(n \log\log n / \log_{|\Sigma|} n)$ bits. The index is optimal in space, apart from lower-order terms, achieving asymptotically the empirical entropy of the text (with a multiplicative constant 1). These data structures also have practical significance, as detailed in [12].

An interesting extension, with practical applications related to image matching, is to develop a data structure that achieves similar space bounds as the 1-D case and the same time bounds as known multidimensional data structures. Multidimensional data present a new challenge when trying to capture entropy, as now the critical notion of *spatial information* also enters into play. (In a strict sense, this information was always present, but we can anticipate more dependence upon spatially linked data.) Stronger notions of compression are applicable, yet the searches are more complicated. Achieving both, is again, a challenge.

Multidimensional Matching

We define a text matrix $T^{(d)}$ as a hypercube in d dimensions with length n, where each symbol is drawn from the alphabet $\Sigma = \{0, 1, \ldots, \sigma\}$. For example, $T^{(2)}$ represents an $n \times n$ text matrix, and $T^{(1)} = T$ simply represents a text document with n symbols.

Handling high-order entropy (and other entropy notions) for multidimensional data in a practical way is difficult. We generalize the notion of hth order entropy as follows. For a given text $T^{(d)}$, we define $H_h^{(d)}$ as

$$H_h^{(d)} = \sum_{x \in A^{(d)}} \sum_{y \in \Sigma} -\operatorname{Prob}[y, x] \cdot \log \operatorname{Prob}[y|x],$$

where $A^{(d)}$ is a d-dimensional text matrix with length h.

A common method used to treat data more contextually (and thus, consider spatial information explicitly) is to *linearize* the data. Linearization is the task of performing somewhat of a "map" to the 1-D case (so that the data is again laid out as we are accustomed to). One technique is described in [15, 16]. Linearization is primarily performed to meet the constraints put forth by Giancarlo [9, 10] in order to support pattern matching in 2-D. (These constraints are readily generalized to multidimensions.)

One major goal of ours in multidimensional matching is to improve the space requirement, without affecting the search times already achieved in literature. Not considering space-efficient solutions (which are absent from current literature), the 2-D pattern matching problem is widely studied by Amir, Benson, Farach, and other researchers [2, 4, 6, 5, 3]. In particular, Amir and Benson [1] give compressed matching algorithms for 2-dimensional cases; however, their pattern matching is not indexing and it needs the scan over entire compressed text.

Suffix arrays and trees have been generalized to multiple dimensions, and a great deal of literature is available that describes various incarnations of these data structures [15, 16], but the vast majority of them discuss just the construction time of these powerful structures. Little work has been done on space-efficient versions of these structures, nor has any real emphasis been given to achieving optimal performance. The hurdles are far more basic than that.

The primary difficulty stems from the fact that there is no clear multidimensional analogue for the Burrows-Wheeler transform (BWT) that still allows for multidimensional pattern matching. The BWT is critical to achieving high-order entropy in one dimension [13, 7, 8, 11, 12]; there, each suffix of the text is sorted and can be indexed using a variety of tricks [7, 8, 11]. Even with just two dimensions, the problem becomes difficult to solve.

In order to support multidimensional pattern matching, the data should be considered from a localized view of the data, namely in terms of hypercubes (which in 1-D is simply

a contiguous sequence of symbols) starting at each position of the text. However, a BWT cannot be formed explicitly upon such a view, as any such localized view violates the critical invariant that suffixes must overlap perfectly. Nevertheless, some basic notions have been explored [9, 10, 15, 16] as a first step in tackling these limitations.

Goals of Study

We hope to make major inroads beyond [15, 16] by developing the crucial notion of a multidimensional BWT. We study the 2D case to highlight combinatorial properties that do not appear in the 1D case, as a first step towards developing a general multidimensional framework. In particular, we are considering a series of novel transformations of the data that simultaneously allow fast access to the data, ease of compression, and do not violate the various constraints proposed by [10]. We then hope to apply it to build a multidimensional suffix array while still retaining the best-known performance bounds (both theoretically and in practice). In addition, much of the literature only discusses extensions to 2-D. We hope to develop data structures that operate for any dimension d and address these two problems:

1. Is there a multidimensional analogue to the Burrows-Wheeler transform captures spatial information and still allows multidimensional pattern matching?
2. Is it possible to achieve a multidimensional suffix array that operates on d-dimensional data in just $n^d H_h + o(n^d)$ bits with $O(\text{polylog} n^d)$ time?

References

[1] A. Amir and G. Benson. Efficient Two-Dimensional Compressed Matching. *DCC*, 1992, 279–288.

[2] A. Amir and E. Porat and M. Lewenstein. Approximate subset matching with Don't Cares. *SODA*, 2001, 305–306.

[3] A. Amir and M. Farach. Efficient 2-Dimensional Approximate Matching of of Half-rectangular Figures. Info. and Computation, 118(1):1–11, 1995.

[4] A. Amir and M. Lewenstein and E. Porat. Faster algorithms for string matching with k mismatches. *SODA*, 2000, 794–803.

[5] A. Amir and G. Benson and M. Farach. Let Sleeping Files Lie: Pattern Matching in Z-Compressed Files. *SODA*, 1994.

[6] A. Amir and G. Benson and M. Farach. Optimal Two-Dimensional Compressed Matching. *ICALP*, 1994, 215–226.

[7] P. Ferragina and G. Manzini. Opportunistic Data Structures with Applications. *FOCS*, 2000.

[8] P. Ferragina and G. Manzini. An Experimental Study of an Opportunistic Index. *SODA*, 2001.

[9] R. Giancarlo and R. Grossi. Multi-Dimensional Pattern Matching with Dimensional Wildcards. *CPM*, 1995, 90–101.

[10] R. Giancarlo and R. Grossi. On the construction of classes of of suffix trees for square matrices: algorithms and applications. Info. and Computation, 130(2):151–182, 1996.

[11] R. Grossi and A. Gupta and J. S. Vitter. High-Order Entropy-Compressed Text Indexes. *SODA*, 2003.

[12] R. Grossi and A. Gupta and J. S. Vitter. When Indexing Equals Compression: Experiments on Suffix Arrays and Trees. *SODA*, 2004.

[13] R. Grossi and J. S. Vitter. Compressed suffix arrays and suffix trees with applications to text indexing and string matching. *STOC*, 2000, 397–406.

[14] D. Gusfield. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology Cambridge University Press, 1997.

[15] D. Kim and Y. Kim and K. Park. Constructing Suffix Arrays for Multi-dimensional Matrices. *CPM*, 1998, 126–139.

[16] D. Kim and Y. Kim and K. Park. Generalizations of suffix arrays to multi-dimensional matrices. *TCS*, 2003.

[17] G. Navarro. The LZ-index: A Text Index Based on the Ziv-Lempel Trie. Manuscript.

[18] K. Sadakane. Compressed text databases with efficient query algorithms based on the compressed suffix array. *ISAAC*, 2000, 410–421.

[19] K. Sadakane. Succinct representations of *lcp* information and improvements in the compressed suffix arrays. *SODA*, 2002.