On the Construction of Classes of Suffix Trees for Square Matrices: Algorithms and Applications

Raffaele Giancarlo¹* and Roberto Grossi²**

¹ Dipartimento di Matematica ed Applicazioni, Università di Palermo, Italy
² Dipartimento di Sistemi e Informatica, Università di Firenze, Italy

Abstract. Given an $n \times n$ matrix with entries defined over an ordered alphabet $\Sigma$, we introduce $4^{n-1}$ classes of index data structures for $TEXT$. Those indices are informally the two-dimensional analog of the suffix tree of a string [15], allowing on-line searches and statistics to be performed on $TEXT$. We provide one simple algorithm that efficiently builds any chosen index in those classes in $O(n^2 \log n)$ worst case time using $O(n^2)$ space. The algorithm can be modified to require optimal $O(n^2)$ expected time for bounded $\Sigma$.

1 Introduction

The development of a uniform framework in which to cast the study of a class of data structures seems to be a worthwhile task since it usually leads to a better understanding of the class and to the design of better algorithms for the construction of the data structures in that class. For instance, the dichromatic framework developed by Guibas and Sedgewick [9] for balanced search trees led to new balanced search tree schemes as well as to new interesting implementations of the algorithms for the management of B-trees [3] and of AVL trees [9].

In this abstract, we provide a uniform framework for the study of index data structures for a two-dimensional matrix $TEXT$, i.e., data structures that are the two-dimensional analog of the suffix tree for a string [15]. Informally, given an $n \times n$ matrix $TEXT$ (referred to as the text) with entries defined over an ordered alphabet $\Sigma$, an index for the text is a tree compactly representing all square submatrices of $TEXT$. Two kinds of queries must be supported by the index: (a) statistical information about the text, for instance, find the largest repeated square submatrix in $TEXT$; (b) on-line pattern matching for an $m \times m$ pattern matrix $PAT$, in which all of the occurrences of $PAT$ as a submatrix of $TEXT$ should be found in time that is independent of the size $n^2$ of $TEXT$. The study

* The work of this author is partially supported by MURST Grant “Algoritmi, Modelli di Calcolo e Strutture Informative”. This work was done while the author was with AT&T Bell Laboratories, Murray Hill, NJ. U.S.A. Email: raffaele@altair.math.unipa.it

** Work supported by MURST of Italy. Part of this work was done while the author was visiting AT&T Bell Laboratories. Email: grossi@di.unipi.it
of indices is motivated by many important applications in low level image processing [16] and compression [18], pattern recognition [11], visual databases [10] and geographic information systems [14]. We obtain:

(1) $4^{n-1}$ classes of indices for $TEXT$. Each class contains $\Pi_{i=2}^{n+1}(2i - 1)!$ indices that we show to be isomorphic, i.e., all indices in a class have the same topological structure. Every index is associated with a different representation of an $n \times n$ matrix as a string. We provide a concise description for each index, in the form of string of $n$ "instructions" to build it.

(2) One simple algorithm that takes in input the matrix $TEXT$ and, as a parameter, the concise description of an index $I_{TEXT}$ from one class in (1). It builds $I_{TEXT}$ in $O(n^2 \log n)$ worst case time using $O(n^2)$ space. Such an algorithm can be seen as a "compiler" for the indices defined in (1).

(3) One simple algorithm, a nonobvious variation of the one in (2), which requires $O(n^2)$ expected time. The probabilistic assumptions are extensions of the ones in [19] and seem to be very mild and realistic.

(4) Each index in a class defined in (1) supports a wide variety of statistical queries about $TEXT$, many of which can be answered in optimal time. Moreover, we have one pattern matching algorithm that answers the on-line query for $PAT$ in $O(m^2 \log |\Sigma| + occ)$ worst case time, where $occ$ is the number of occurrences of $PAT$ in $TEXT$. It can be seen as a general pattern matcher for the indices defined in (1).

We have the following remarks:

- The general problem of building an index representing all submatrices of an arbitrary $n \times w$ matrix $TEXT$ has been considered in [4, 6, 8]. The index proposed in [8] can be built in $O(\max(n,w)^2 \min(n,w)^3)$ worst case time, while the ones proposed in [4] and [6] can be built in $O(\max(n,w) \min(n,w)^2 \log(nw))$ worst case time. All three indices support a wide variety of queries (see [4, 6, 8]). Moreover, it has been shown in [4] that any index representing compactly all submatrices of $TEXT$ must require $\Omega(\max(n,w) \min(n,w)^2)$ space, while an index representing only the square submatrices of $TEXT$ can be stored in $O(nw)$ space. It is worth pointing out that many images and maps are represented and queried as square matrices (e.g., see [17]).

- Up to date, the known indices for square matrices are the PAT-tree by Gonnet [8], the $Lsuffix$ tree by Giancarlo [5] and the two dimensional suffix trie by Storer [18]. We show that they are members of two classes defined in (1). Moreover, we provide $4^{n-1} - 2$ new classes of indices. That is important from the applications point of view since much of the on-going research in spatial data structures tries to understand which linear representation of a square matrix best captures a given set of neighborhood properties of the given matrix [14]. Since each index defined in (1) corresponds to a distinct linear representation of a matrix, we provide the freedom to choose from a large number of linear representations for the construction of indices, guaranteeing efficient algorithms (for the worst and expected case) for each index.

- From the algorithmic viewpoint, the framework from which we derive the indices in (1) allows us to blend together, improve and generalize in a nonobvious