Theoretical Computer Science 647 (2016) 74–84

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Indexing and querying color sets of images

Djamal Belazzougui [a], Roman Kolpakov [b], Mathieu Raffinot [c,*]

[a] DTISI, Research Centre for Scientific and Technical Information (CERIST), Algeria
[b] Lomonosov Moscow State University, Moscow, Russia
[c] LaBRI, Université de Bordeaux 1, France

ARTICLE INFO

Article history:
Received 16 December 2014
Received in revised form 28 July 2016
Accepted 31 July 2016
Available online 10 August 2016
Communicated by G. Ausiello

Keywords:
Image algorithms
Text algorithms
Fingerprint
Set of characters
Set of colors
Maximal locations

ABSTRACT

We aim to study the set of color sets of continuous regions of an image given as a matrix of m rows over $n \geq m$ columns where each element in the matrix is an integer from $[1, \sigma]$ named a *color*. The set of distinct colors in a region is called fingerprint. We aim to compute, index and query the fingerprints of all rectangular regions named rectangles. The set of all such fingerprints is denoted by $\mathcal{F}$. A rectangle is *maximal* if it is not contained in a greater rectangle with the same fingerprint. The set of all locations of maximal rectangles is denoted by $\mathcal{L}$. We first explain how to determine all the $|\mathcal{L}|$ maximal locations with their fingerprints in expected time $O(nm^2\sigma)$ using a Monte Carlo algorithm (with polynomially small probability of error) or within deterministic $O(nm^2\sigma \log(\frac{|\mathcal{L}|}{nm^2}+2))$ time. We then show how to build a data structure which occupies $O(nm \log n + |\mathcal{L}|)$ space such that a query which asks for all the maximal locations with a given fingerprint f can be answered in time $O(|f| + \log\log n + k)$, where k is the number of maximal locations with fingerprint f. If the query asks only for the presence of the fingerprint, then the space usage becomes $O(nm \log n + |\mathcal{F}|)$ while the query time becomes $O(|f| + \log\log n)$. We eventually consider the special case of squared regions (squares).

1. Introduction

In this paper, we are interested in studying and indexing the set of all sets of distinct colors of continuous regions of a given image in order to quickly answer to several queries on this set, for instance, does there exist at least one region with a given set of colors and if so, what are the positions of all such regions in the image?

Also the considered indexing structures and algorithms are a good approach towards efficient image comparisons and clustering based on color sets. To our knowledge, this is the first time such a problem is formalized and analyzed, and to build a general well founded algorithmic framework we begin by formalizing our notions.

We consider an image as a matrix M of m rows over n columns (see Fig. 1) where each $a_{i,j}$ is an integer from $[1, \sigma]$.[1] This matrix is called an image and henceforth, we will refer to the integers stored in the matrix by colors or characters.

To design our indexing structures and algorithms, we extend the definition of fingerprints (or sets of distinct characters), initially defined on sequences [1,6,7,12], to images. While classical pattern matching approaches on images have already

* Corresponding author.
E-mail address: mathieu.raffinot@labri.fr (M. Raffinot).

[1] We assume that $\sigma \leq nm$. If this is not the case, then we can build a dictionary data structure that occupies $O(nm)$ space and that can remap in constant time distinct characters from the original alphabet to distinct integers in $[1, nm]$.

http://dx.doi.org/10.1016/j.tcs.2016.07.041
0304-3975/

$$\begin{array}{ccccc} a_{m;1} & a_{m;2} & \cdots & a_{m;n-1} & a_{m;n} \\ a_{m-1;1} & a_{m-1;2} & \cdots & a_{m-1;n-1} & a_{m-1;n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{2;1} & a_{2;2} & \cdots & a_{2;n-1} & a_{2;n} \\ a_{1;1} & a_{1;2} & \cdots & a_{1;n-1} & a_{1;n} \end{array}$$

Fig. 1. An image.

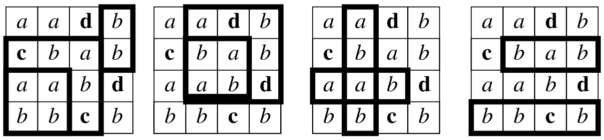

Fig. 2. Example of maximal rectangles (bold frontiers) in a small image.

been studied [2–4], this paper is the first (to our knowledge) to focus on the character sets. We assume below that $m \le n$. We denote by $\langle i_0, i_1; j_0, j_1 \rangle$ where $i_0 \le i_1$ and $j_0 \le j_1$, the rectangle in M bounded by the i_0-th and i_1-th rows and j_0-th and j_1-th columns including these rows and columns. We also denote by $f\langle i_0, i_1; j_0, j_1 \rangle$ the set of distinct colors contained in the rectangle $\langle i_0, i_1; j_0, j_1 \rangle$. This set is called the *fingerprint* of $\langle i_0, i_1; j_0, j_1 \rangle$.

Definition 1. A rectangle in an image is *maximal* if it is not contained in a greater rectangle with the same fingerprint.

In other words a rectangle is maximal if any extension of the rectangle in one of the four directions will add at least one color not present in the rectangle. Fig. 2 shows an example of an image containing many maximal rectangles.

In this article, as a first step toward our general goal of indexing all color sets of continuous regions, we focus on indexing and answering queries on fingerprints of all maximal rectangles of the input image. Given a fingerprint f, a maximal rectangle with the fingerprint f is called a *maximal location* of f. We denote by $\mathcal{L}$ the set of all maximal locations, and by $\mathcal{F}$ the set of all distinct fingerprints of rectangles in the image. It is easy to see that $|\mathcal{F}| \le |\mathcal{L}| \le nm^2\sigma$. All our results assume the standard RAM model with word size $w = \Omega(\log(n+\sigma))$ and with all standard arithmetic and logic operations (including multiplication) taking constant time. We prove below the following theorems:

Theorem 1. *Given an image of m rows by $n \ge m$ columns, we can determine all the $|\mathcal{L}|$ maximal locations with their fingerprints in expected time $O(nm^2\sigma)$ with a Monte Carlo algorithm (with polynomially small probability of error) or within deterministic $O(nm^2\sigma \log(\frac{|\mathcal{L}|}{nm^2}+2))$ time.*

Note that the total deterministic time is $O(nm^2\sigma \log \sigma)$ in the worst case (when we have $|\mathcal{L}| = \Theta(nm^2\sigma)$), but is only $O(nm^2\sigma)$ (which is as good as the Monte Carlo time) as long as $|\mathcal{L}| \le O(nm^2)$.

Theorem 2. *Given an image of m rows by $n \ge m$ columns, whose set $\mathcal{L}$ of all maximal locations and set $\mathcal{F}$ of associated fingerprints have been already determined, we can build a data structure which occupies space $O(nm \log n + |\mathcal{L}|)$ such that a query which asks for all the maximal locations with a given fingerprint f can be answered in time $O(|f| + \log\log n + k)$, where k is the number of maximal locations with fingerprint f. If the query asks only for the presence of the fingerprint, then the space usage becomes $O(nm \log n + |\mathcal{F}|)$ while the query time becomes $O(|f| + \log\log n)$.*

The construction times of the data structures mentioned in Theorem 2 are respectively $O(nm \log n \log\log n + |\mathcal{L}|)$ and $O(nm \log n \log\log n + |\mathcal{F}|)$.

We eventually consider the special case in which only squared regions called squares are considered instead of rectangles. We develop a specialized faster algorithm for this case.

1.1. Notations and tools

Let $\langle i_0, i_1; j_0, j_1 \rangle$ be a rectangle. For this rectangle, the i_0-th row is called the bottom row, the i_1-th row is called the top row, the j_0-th column is called the left column, and the j_1-th column is called the right column. We also define the following notions derived from Definition 1: a rectangle $\langle i_0, i_1; j_0, j_1 \rangle$ is *maximal to the left* (resp. *to the right*) if rectangle $\langle i_0, i_1; j_0 - 1, j_1 \rangle$ (resp. $\langle i_0, i_1; j_0, j_1 + 1 \rangle$) doesn't have the same fingerprint, and is *maximal to the bottom* (resp. *to the top*) if rectangle $\langle i_0 - 1, i_1; j_0, j_1 \rangle$ (resp. $\langle i_0, i_1 + 1; j_0, j_1 \rangle$) doesn't have the same fingerprint. It is obvious that a rectangle is maximal if and only if it is maximal in all the directions. One of our solutions uses the following lemma by Muthukrishnan [14]:

Lemma 1. *Given a sequence of colors $T[1, n]$ each chosen from the same alphabet of colors $[1, \sigma]$, in time $O(n)$ we can preprocess the sequence into a data structure which occupies $O(n)$ space so that given any range $[i, j]$ we can find the set of all distinct colors*