A-36

Nordic Journal of Computing

COMPLEXITY ASPECTS OF TWO-DIMENSIONAL DATA COMPRESSION *

Hans L. Bodlaender†
Department of Computer Science, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands
HANSB@CS.RUU.NL

Teofilo F. Gonzalez
Department of Computer Science, University of California,
Santa Barbara, CA 93106-5110, USA‡

Ton Kloks§
Department of Mathematics and Computing Science,
Eindhoven University of Technology,
P. O. Box 513, 5600 MB Eindhoven, the Netherlands.

Abstract. Let M be a 2-dimensional colored map which has been digitized into a large 2-dimensional array (M). We define a class of languages (called *rectilinear*) to describe our digitized maps and classify them based on their level of succinct representation. We also study the map compression problem, i.e., the problem of finding for any given map a shortest description within a given language. For one dimensional maps we show that a shortest description can be generated quickly for some languages, but for other languages the problem is NP-hard. We also show that a large number of linear time algorithms for our languages generate map descriptions whose length is at most twice the length of the minimum length description. For all our languages we show that the two dimensional map compression problem is NP-hard. Furthermore, for one of the most succinct of our languages we present evidence suggesting that finding a near-optimal map compression is as difficult as finding an optimal compression.

1. Introduction

Let M be a 2-dimensional colored map, e.g., a landscape, to be stored in a digital computer system and/or to be drawn on a terminal screen. Assume that the map has been digitized into a large 2-dimensional array (M). I.e., a

* A preliminary version of this paper appeared in the Proceedings of the Data Compression Conference, April 1991, pp 287 - 296.

† The work of this author was partially supported by the ESPRIT II Basic Research Actions Program of the EC, under contract No. 3075 (project ALCOM).

‡ TEO@CS.UCSB.EDU

§ The work of this author was supported by the Foundation for Computer Science (S.I.O.N.) of the Netherlands Organization for Scientific Research (N.W.O.).

Received September 1995.

large uniform square grid partitions the map into n by m small grid squares denoted by $I_{n,m}$. Grid square $I_{i,j}$ is associated with the matrix entry (i,j) in M. Each matrix entry $(M(i,j))$ is assigned an integer $l \in [0,p)$ to denote the representative color for grid square $I_{i,j}$. Hereafter we refer to matrix entries and grid squares interchangeably.

In many practical applications a map contains large singly colored regions, and also regions in which the colors change rapidly. So, finding a good probabilistic model that represents the distribution of the different colors is at least difficult, if not impossible.

In this paper we define "languages" to describe our digitized maps. The objective is to find a shortest description within a given language. For example, instead of describing a 2-dimensional digitized map by its corresponding matrix we describe it by an *rp-compression*, i.e., collection of tuples of the form (RP_i, c_i), where RP_i is a subset of grid squares bordered by a simple rectilinear polygon without holes and c_i is a color (i.e., an integer value in the range [0,p)). An rp-compression represents map M if map M is generated by starting from a grid without colors assigned to the entries and then coloring all the grid squares in RP_1 with color c_1; then all the ones in RP_2 with color c_2, and so forth. Note that if a grid square is in two or more rectilinear polygons its final color is the last one assigned to it. We shall refer to these rectilinear polygons as *c-rectilinear polygons* and denote the language just defined L_{2RP}. An rp-compression is said to be an *pdrp-compression* if all the c-rectilinear polygons in it are pairwise disjoint. The language L_{2PDRP} is obtained by replacing rp-compression by pdrp-compression in the definition of L_{2RP}. We say that the amount of information required to represent a map under L_{2RP} (L_{2PDRP}) is the total number of corners of the c-rectilinear polygons in the rp-compression (pdrp-compression).

Maps usually have a more succinct representation under language L_{2RP} than under language L_{2PDRP}. The following example shows the case when there is a dramatic difference between the minimum length representation of a map in these two languages. The "map" is given in a Fig. 1 and it consists of $n \times n$ grid squares, each colored black (represented by a shaded square) or white (represented by a blank). All grid squares are white except for grid squares (i,j) for all i and j even. The smallest description under L_{2PDRP} for the map given in Fig. 1 contains at least $\Omega(n^2)$ black squares (which are islands in the white area). But, under language L_{2RP} the map can be described by one large black square followed by $O(n)$ white strips. Remember that a c-rectilinear polygon in an rp-compression does not have to border exactly an area of a given color.

Let M be a map and L be any language. We use $T(L,M)$ to denote the number of "values" in a minimum length representation of M in L. Two languages L and L' are said to be *equivalent* if for every map M, $T(L,M)$ is $O(T(L',M))$ and $T(L',M)$ is $O(T(L,M))$. A language L is said to be *more succinct* than language L' if for every map M, $T(L,M)$ is $O(T(L',M))$, but $T(L',M)$ is not $O(T(L,M))$. One can argue that our classification scheme is not fair because we do not take into account the maximum number of

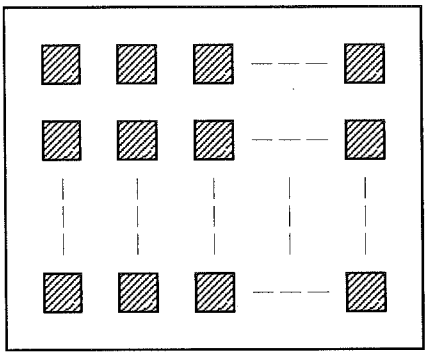

Fig. 1: Example.

bits in each of the values. So extreme care must be taken when classifying languages because our comparison holds only in certain domains. From the above discussion it follows that L_{2RP} is more succinct than L_{2PDRP}. Also, L_{2PDRP} is more succinct than L, where L is the language that represents a map by its n by n matrix of colors. Note that this last comparison is not a fair one because each value in the matrix is an integer value in the range [0,p), where as in the representation for the other language the values are integers in the range [0,n). Hereafter we concentrate on languages in which the c-rectilinear polygons may overlap.

An rp-compression is said to be an *rp-nr-compression* if all the c-rectilinear polygons in it assigned the same color are adjacent in the description. A restricted version of L_{2RP} is the language $L_{2RP\text{–}NR}$ obtained by replacing rp-compressions by rp-nr-compressions. The *NR* stands for *no recoloration* because our procedure that generates M from the rp-nr-compression has the property that once a grid square has been colored with its correct color, it will never be colored with another color different from its correct color. This is not true for rp-compressions. By definition $T(L_{2RP}, M)$ is $O(T(L_{2RP\text{–}NR}, M))$. However, it is not possible to show that $T(L_{2RP\text{–}NR}, M)$ is $O(T(L_{2RP}, M))$ for every map M. An optimal rp-compression for the map given in Fig. 2 under language L_{2RP} has $O(k)$ corners, whereas under language $L_{2RP\text{–}NR}$ all rp-nr-compressions have $\Omega(k^2)$ corners.

In this paper we study the $2CR_{RP}$ ($2CNR_{RP}$) map compression problem defined as the problem of finding a minimum length representation for a 2-dimensional map under the L_{2RP} ($L_{2RP\text{–}NR}$) language. When the c-rectilinear polygons have exactly four sides (called *c-rectangles*) the above problems are referred to as the $2CR_R$ and the $2CNR_R$ problem. In this case the objective function reduces to minimizing the number of rectangles in the description and we use the term *r-compression* (*r-nr-compression*) instead of rp-compression (rp-nr-compression). The languages are referred to as L_{2R} and $L_{2R\text{–}NR}$, respectively. In each of these two cases the rectangular languages are equivalent (with respect to succinctness) to their corresponding

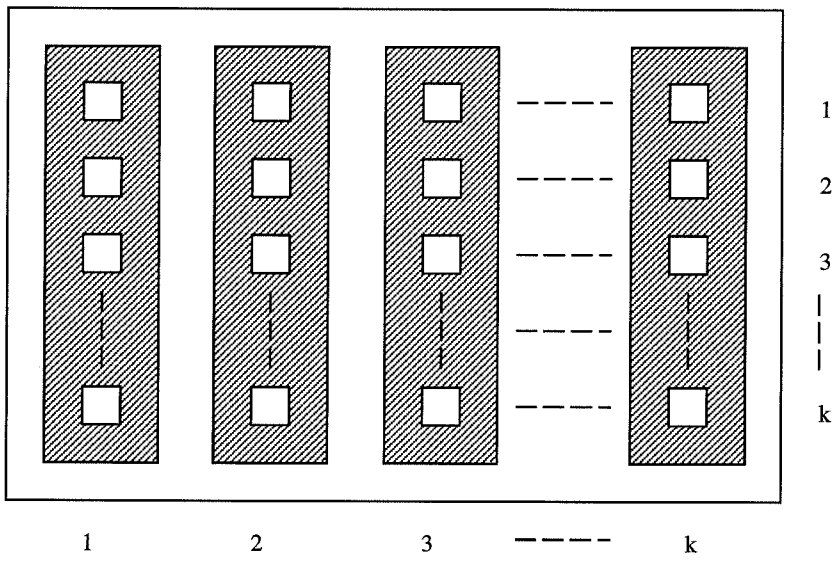

Fig. 2: Example.

rectilinear polygon language. The reasoning for this is that any rectilinear polygon with k corners may be covered by at most $2k$ rectangles (the fastest procedure to accomplish this is given in [10]) and a rectangle is a rectilinear polygon.

When the two dimensional matrix representing a map has a single row, the map is said to be one dimensional and the names for the above problems and languages are prefixed by a one instead of a two. Note that in this case the $1CR_{RP}$ ($1CNR_{RP}$) is identical to the $1CR_R$ ($1CNR_R$) problem because all c-rectilinear polygons may be replaced simply by c-rectangles. All of these one-dimensional languages are equivalent with respect to succinctness. The reason for this is that every r-nr-compression is also an r-compression and thus $T(M, L_{1R})$ is $O(T(M, L_{1R-NR}))$. The proof of the converse follows from a lower bound for the number of c-rectangles in an optimal compression established in the proof of Theorems 3 and 4.

A restricted version of these problems, referred to by $\alpha R\beta_\gamma$, where $\alpha \in \{1, 2\}$, $\beta \in \{CR, CNR\}$ and $\gamma \in \{R, RP\}$, is the $\alpha\beta_\gamma$ problem in which each color appears in at most two entries in M. Later on it will be evident why we introduced these versions of our problems.

In many practical situations the number of different colors in M is small. Because of this we shall also investigate the complexity of the $\alpha\beta_\gamma(k)$ problem, where $\alpha \in \{1, 2\}$, $\beta \in \{CR, CNR, RCR, RCNR\}$ and $\gamma \in \{R, RP\}$, is the $\alpha\beta_\gamma$ problem in which the number of different colors in M is bounded by the constant k (which is not an input parameter).

In section 2 we consider the one dimensional case. We present an $O(n^3)$ time dynamic programming algorithm for the $1CR_R$, where n is the size of

array M. For the no recoloration variant, we show that the $1CNR_R$ problem is an NP-hard problem. For both of these problems we show that any algorithm that avoids a set of "bad decisions" generates a solution with a number of c-rectangles that is within two times the number of c-rectangles in an optimal solution. We also show that for both problems a solution within two times optimal can be generated in linear time by a large number of algorithms. We also show that problems $1RCR_R$ and $1RCNR_R$ reduce to a well known graph problem which can be solved efficiently.

In section 3, we prove that for the two dimensional case, even restricted versions of our problems are NP-hard. Specifically, we show that the $2RCR_R$ and $2RCNR_R$ problems are NP-hard. Obviously, this also shows that the $2CR_R$, and $2CNR_R$ are also NP-hard. Our reduction can be easily modified to show that the $2CR_{RP}$, and $2CNR_{RP}$ problems are also NP-hard. In section 4, we show that the map compression problem remains NP-hard even when the number of different colors in M is bounded by a small constant ($2CR_R(4)$ and $2CNR_R(2)$). We discuss some generalizations of our results and pose some interesting open problems in section 5.

There are many image compression techniques available in the literature (see [7]). For brevity we cannot elaborate on all of these techniques. At present the JPEG compression technique is popular. JPEG is based on applying two-dimensional discrete cosine transform (DCT) to 8 by 8 pixel subimages, then quantizing the DCT coefficients, and stretching them into a one dimensional array that is encoded using Huffman and run-length coding.

Quad-tree decomposition methods partition the image into four subimages by introducing a vertical and a horizontal cut. The subimages are partitioned recursively until the resulting images are "uniform". It is simple to show that this type of compression technique, when applied to produce lossless compressions, is a restricted form of partition into disjoint rectangles (which is one of the compression methods we study). Therefore, the quad-tree representation is not as succinct as that for general rectangle decomposition.

In pyramid decompositions an image is transformed into a smaller "blurry" subimage plus other (normally three) subimages containing "edge information" in such a way that the original image can be generated (either exactly or approximately) from these subimages. The "blurry" subimage is decomposed recursively and all the "blurry" subimages generated can be used for browsing. All the subimages in the decomposition are then compressed. Since the subimages containing "edge information" are fairly "uniform", one could use heuristics that decompose the subimages into rectangles and then compress the resulting representation. Thus the analysis in this paper is not only useful for images, but it is also useful for images that arise in other decomposition methods, like in pyramids, wavelets, etc.

2. One dimensional maps

In this section we consider the one dimensional map compression problem with and without recoloration. We present an $O(n^3)$ time dynamic programming algorithm for the $1CR_R$, where n is the number of entries in M. However, for the no recoloration variant ($1CNR_R$) we show it is an NP-hard problem. For both of these problems we show that any algorithm that avoids a set of "bad decisions" generates a solution with a number of c-rectangles that is at most two times the number of c-rectangles in an optimal solution. We also show that for both problems a solution within two times optimal can be generated in linear time. In subsection 2.4 we show that problems $1RCR_R$ and $1RCNR_R$ reduce to a well known graph problem which can be solved efficiently.

An instance of our 1-dimensional problems is represented by $INS = (M, n, p)$, where $M_i \in [0, p)$ for $1 \leq i \leq n$. When we refer to a *compression* we mean either an r-compression or an r-nr-compression. Let R_l be a c-rectangle in compression $R = (R_1, R_2, \ldots, R_r)$. We shall refer to the n entries in array M as array elements (map grid squares or simply elements). Array element i included in R_l is said to be *tight* if it is not included in $R_{l+1}, \ldots, R_r$. Since R is a compression, we know that if array element i is tight in R_l, then $c_l = M_i$. A c-rectangle is said to be *tight* if the rightmost and leftmost elements in it are tight, and a compression is said to be *tight* if all its c-rectangles are tight.

2.1 Algorithm for the $1CR_R$ problem

Let $INS = (M, n, p)$ be any instance of the $1CR_R$ problem. For $1 \leq i \leq j \leq n$, we use $INS_{i,j}$ to represent the subinstance of the $1CR_R$ problem INS defined over array elements $(i, i+1, \ldots, j-1, j)$. Let $g(i, j)$ denote the minimum number of c-rectangles in an optimal solution for the instance $INS_{i,j}$ of the $1CR_R$ problem. Obviously, $g(i, i) = 1$ for $1 \leq i \leq n$. Let $R = (R_1, R_2, \ldots, R_r)$ be an optimal r-compression for $INS_{i,j}$. In Lemma 1 we prove an important property of optimal r-compressions for any instance $INS_{i,j}$ of the $1CR_R$ problem. This will aid us in the development of the $O(n^3)$ time algorithm that generates optimal r-compressions.

LEMMA 1. *Every instance $INS_{i,j}$ of the $1CR_R$ problem has an optimal r-compression that is tight. Furthermore, element i is tight in c-rectangle R_1.*

PROOF. We prove a stronger result. That is, given any r-compression for $INS_{i,j}$ it can be transformed without increasing the number of c-rectangles into an r-compression for $INS_{i,j}$ that satisfies the conditions of the lemma.

First we show that any r-compression R'' for any instance $INS_{i,j}$ of the $1CR_R$ problem can be transformed into a tight r-compression R' for $INS_{i,j}$ with no more c-rectangles than the ones in R''. The r-compression R' is constructed from R'' as follows. If R''_l is tight then R'_l is just R''_l; otherwise,

R'_l is R''_l after deleting from each end the elements that are not tight. Delete from R' the c-rectangles with no elements. One can easily show that R' is a tight r-compression for $INS_{i,j}$ with no more c-rectangles than the ones in R''.

If element i is tight in R'_1 of R' then R is just R' and the lemma follows. So assume that element i is tight in R'_l for some $l > 1$. Let k be the rightmost tight element in R'_l. Since the elements $i, i+1, \ldots, k$ cannot be tight in R'_1, $R'_2, \ldots$ R'_{l-1}, there are no elements to the left of i in $INS_{i,j}$, and R' is tight, it must be that elements $i, i+1, \ldots, k$ are not included in the rectangles $R'_1, R'_2, \ldots, R'_{l-1}$. Let $(R_1, R_2, R_3, \ldots, R_l, R_{l+1}, \ldots, R_r)$ be $(R'_l, R'_1, R'_2, \ldots, R'_{l-1}, R'_{l+1}, \ldots, R'_r)$. Clearly, element i is tight in R_1, R is a tight r-compression for $INS_{i,j}$ and there are no more c-rectangles in R than in R'. So R satisfies the conditions of the lemma. Therefore, any instance $INS_{i,j}$ of the $1CR_R$ problem has an optimal r-compression that satisfies the conditions of the lemma. This completes the proof of the lemma. □

Let $i_1 < i_2 < \ldots < i_s$ be all the elements in $(i, i+1, \ldots, j-1, j)$ colored M_i. Let R be an optimal r-compression for $INS_{i,j}$ that satisfies the conditions of Lemma 1. Let $j_1 < j_2 < \ldots < j_q$ be the tight elements in R_1. From the conditions of Lemma 1 we know that $q \geq 1$ and $j_1 = i$. By the principle of optimality it is simple to show that if $q = 1$, then $g(i,j) = g(j_1+1, j)+1$; and if $q > 1$, then $g(i,j) = g(j_1+1, j_2-1)+g(j_2, j)$, where $g(k,l) = 0$ when $k > l$. Therefore, $g(i,j)$ can be computed via dynamic programming techniques as follows. Let $g(i,i) = 1$, for $1 \leq i \leq n$; let $g(i,j) = 0$, for $i > j$; and for $i < j$ define

$$g(i,j) = min\{g(i_1+1, j)+1; min_{1<k\leq s}\{g(i_1+1, i_k-1)+g(i_k, j)\}\}.$$

We define procedure *DP* to compute the $g(i,j)'s$ for all $j - i = 1$, then $2, 3, \ldots$, until $n-1$, by using the above recursive formulation. It is simple to show that procedure *DP* takes $O(n^3)$ time to find the number of c-rectangles in an optimal r-compression for any instance of the $1CR_R$ problem. By using standard techniques one can also generate an optimal r-compression in $O(n^3)$ time. These results are summarized in the following theorem.

THEOREM 1. *Procedure DP generates an optimal r-compression in $O(n^3)$ time for any instance, $INS = (M, n, p)$, of the $1CR_R$ problem.*

PROOF. By the above discussion. □

2.2 The $1CNR_R$ problem is NP-hard

Now we show that the $1CNR_R$ problem is *NP*-hard by reducing the feedback edge set (*FES*) problem to it. The *FES* problem, formally defined below, is an *NP*-hard problem [4].

DEFINITION 1. **Feedback edge set (FES):** *Given a directed graph* $G = (V, E)$, *and integer* $0 < k \leq \mid E \mid$, *the FES problem consists of determining whether there is a set* $W \subseteq E$ *with* k *edges such that the graph* $(V, E - W)$ *is acyclic.*

A *linear ordering of a* directed acyclic graph with n vertices is an assignment of all the vertices to the integer points in $[1, n]$ along a straight line (exactly one vertex per point) in such a way that all edges are directed from left to right. Our main result in this subsection is given by the following theorem.

THEOREM 2. *The* $1CNR_R$ *problem is NP-hard.*

PROOF. We prove this theorem by showing that the *NP*-complete problem, *FES*, polynomially reduces to the $1CNR_R$ problem. Given any instance of the feedback edge set problem, *FES*, we construct an instance of the $1CNR_R$ problem, *NR*, as follows. Let (k', G) be the input to the *FES* problem, where k' is any positive integer and $G = (V, E)$ is a directed graph with $n' = \mid V \mid$, and $m' = \mid E \mid$. Label the edges $e_i = (f_i, t_i)$, for $1 \leq i \leq m'$. From *FES* we construct the instance *NR* of the $1CNR_R$ problem with n elements, where $n = (2q+4)m'$ and $q = 2m'+k'$. The number of different colors is $p = qm'+n'$. Colors 1, 2, ..., n' represent the vertices in G and the remaining colors are introduced to enforce a special type of feasible solutions. We partition the set of elements in *NR* into m' consecutive sections $(T_1, T_2, \ldots, T_{m'})$ each with $2q+4$ adjacent elements. The *ith* section (T_i) represents edge e_i and it contains elements $((i-1)(2q+4)+1, (i-1)(2q+4)+2, \ldots, (i)(2q+4))$ which are assigned colors $(q^i, (q-1)^i, \ldots, 2^i, 1^i, f_i, t_i, t_i, f_i, 1^i, 2^i, \ldots, (q-1)^i, q^i)$, where each j^i is a different color. The four elements in the middle of each section, i.e., the ones colored t_i and f_i are called *edge* elements (because they are introduced to represent edges) and the other ones are called *block elements.* For convenience, we say that a c-rectangle with only tight edge (block) elements is called an *edge (block) c-rectangle.*

EXAMPLE 1. The instance *FES* is the graph $G = (V, E)$ given in Fig. 3 and $k' = 1$. The value for $n' = 3$ and $m' = 3$.

The instance *NR* constructed by our procedure has: $q = 7$; $p = 24$; and $n = 54$. The color vector C_{54} is: $7^1, \ldots, 2^1, 1^1, 1, 2, 2, 1, 1^1, 2^1, \ldots, 7^1, 7^2, \ldots, 2^2, 1^2, 2, 3, 3, 2, 1^2, 2^2, \ldots, 7^2, 7^3, \ldots, 2^3, 1^3, 3, 1, 1, 3, 1^3, 2^3, \ldots, 7^3$.

We claim that *FES* has a feedback edge set with cardinality k' if, and only if, *NR* has an r-nr-compression with no more than $w = qm' + 2m' + k'$ c-rectangles. First lets prove that if *FES* has a feedback edge set with cardinality k', then *NR* has an r-nr-compression with at most w c-rectangles. Let W be a feedback edge set with cardinality k' and let $i_1, i_2, \ldots, i_{k'}$ be the indices of the edges in it. For $1 \leq l \leq n$, let j_l be the vertex at integer point l in a linear ordering of the directed acyclic graph with vertex set V and edge set $E - W$. We define the r-nr-compression R as the minimum cardinality

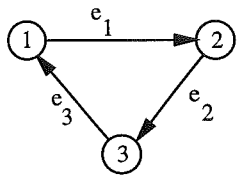

Fig. 3: Graph $G = (V, E)$.

r-nr-compression in which all the c-rectangles with color α appear before the ones with color β whenever α is before β in the following linear ordering of the colors:

$$q^1, \ldots, 2^1, 1^1, q^2, \ldots, 2^2, 1^2, \ldots, q^{m'}, \ldots, 2^{m'}, 1^{m'}, j_1, j_2, \ldots, j_n.$$

Clearly, all the tight block elements appear before all the tight edge elements in R, so it must be that the block c-rectangles appear before the edge c-rectangles in R. It is simple to show that the number of block c-rectangles is qm', i.e., one for each pair of differently colored block elements. Let us now consider the edge c-rectangles. The four edge elements in each section appear in edge c-rectangles that include only edge elements. The edge elements introduced for $e_i = (f_i, t_i)$ are included by exactly two edge c-rectangles if f_i appears before t_i in $(j_1, j_2, \ldots, j_n)$; and exactly three edge c-rectangles, otherwise. Therefore, the edge elements introduced for $e_i \in E - W$ are included in exactly two edge c-rectangles and the ones in W are included in three edge c-rectangles. Therefore, the total number of c-rectangles in R is $w = qm' + 2m' + k'$. Hence, NR has an r-nr-compression with at most w c-rectangles.

Let us now prove that if NR has an r-nr-compression with at most w c-rectangles then FES has a feedback edge set with cardinality k'. Let R be any r-nr-compression for NR with at most w c-rectangles. If edge elements from different sections are tight in the same c-rectangle, then the number of block c-rectangles is at least $(m' + 1)q$ and the number of edge c-rectangles are at least n'. But then

$$(m' + 1)q + n' \geq m'q + 2m' + k' + n' > w,$$

since $q = 2m' + k'$ and $n' \geq 1$. So it cannot be that edge elements from different sections are tight in the same c-rectangle. Therefore, it is possible to reorder the c-rectangles in R so that all the block c-rectangles appear before all the edge c-rectangles. The number of block c-rectangles is at least qm'. Therefore, the number of edge c-rectangles is at most $2m' + k'$. Since the number of edge c-rectangles required for the edge elements in a section is at least two, then at least $m' - k'$ sections have the property that their edge elements are tight in at most two edge c-rectangles. Therefore, for each of these sections the edge c-rectangles colored f_i appear before the

ones colored t_i in the r-nr-compression. Since all the edge c-rectangles in which edge element colored j is tight are adjacent, then let $j_1, j_2, \ldots, j_{n'}$ be the ordering in which the colors of these tight elements appear in the r-nr-compression. It is simple to prove that the ordering of the vertices corresponding to the above ordering of the colors is a linear ordering of G with at most k' edges deleted in which all edges are directed from left to right. Therefore, G has a feedback edge set with cardinality k'. This completes the proof of the theorem. □

2.3 Approximation algorithms for the $1CR_R$ and $1CNR_R$ problems

Let us now consider approximation algorithms for the $1CR_R$ and the $1CNR_R$ problems. We say that a compression (r-compression or an r-nr-compression) is *irreducible* if no two adjacent elements colored with the same color are tight in different c-rectangles. Given a reducible compression one can easily transform it into an irreducible one. An instance of the $1CNR_R$ problem is said to be *c-simple* if every pair of adjacent array elements are colored differently. Given any instance X of the $1CNR_R$ problem we say that the c-simple instance $S(X)$ corresponds to it if $S(X)$ can be obtained from X by collapsing adjacent array elements with the same color into a single array element assigned that color.

The following theorem establishes the fact that any algorithm that generates irreducible r-nr-compressions for an instance *INS* of the $1CNR_R$ problem with $\hat{f}(INS)$ c-rectangles has the property that $\hat{f}(INS)/f^*(INS) < 2$, where $f^*(INS)$ is the number of c-rectangles in an optimal solution for the instance *INS* of the $1CNR_R$ problem. In Theorem 4 we establish an identical bound for the $1CR_R$ problem.

THEOREM 3. *Any algorithm that generates irreducible r-nr-compressions for an instance INS of the $1CNR_R$ problem with $\hat{f}(INS)$ c-rectangles has the property that $\hat{f}(INS)/f^*(INS) < 2$, where $f^*(INS)$ is the number of c-rectangles in an optimal solution for the instance INS of the $1CNR_R$ problem.*

PROOF. We prove the theorem by showing that for any instance of the $1CNR_R$ problem the ratio between the number of c-rectangles in any irreducible cover and the number of c-rectangles in an optimal r-nr-compression is at most two. Let X be any instance of the $1CNR_R$ problem and let $S(X)$ be the c-simple instance equivalent to it. Let n be the number of elements in $S(X)$. Clearly, any algorithm that generates an irreducible r-nr-compression for an instance X of the $1CNR_R$ problem, has a number of c-rectangles that is at most n. Let OPT_n denote the minimum number of c-rectangles in an optimal r-nr-compression for any c-simple instance of the $1CNR_R$ problem with n elements. It is simple to see that we complete the proof of the theorem by showing that $OPT_n \geq (n+1)/2$. The proof is by induction on n. It is obviously true when n is 1. Assume it holds for $1, 2, \ldots, n-1$ and

let us prove it holds for n. Let R be any optimal r-nr-compression for any c-simple instance with n elements. Let $i_1, i_2, \ldots, i_q$ be the tight elements in the first c-rectangle in R. Clearly, the c-rectangles in R after the first one may not contain any of the i_j elements. Therefore, an optimal r-nr-compression has at least as many c-rectangles as the sum of the number of c-rectangles in an optimal solution to the $q+1$ resulting subproblems (by resulting subproblems we mean the elements between 1 and $i_1 - 1, i_1 + 1$ and $i_2 - 1$, and so on) plus one. Let $j_1, j_2, \ldots$ be the number of elements in each of the nonempty resulting subproblems (i.e., each subproblem has at least one element). Therefore, $OPT_n \geq 1 + \sum OPT_{j_k}$ and by the induction hypothesis we know that this quantity is at least $1 + \sum((1 + j_k)/2)$. Since $q + \sum j_k = n$ and since the leftmost and the rightmost subproblems may have zero elements, then there are at least $q - 1$ nonempty subproblems. Therefore, $OPT_n \geq (n+1)/2$. This completes the proof of the theorem. □

THEOREM 4. *Any algorithm that generate irreducible r-nr-compressions for an instance INS of the* $1CR_R$ *problem with* $\hat{f}(INS)$ *c-rectangles has the property that* $\hat{f}(INS)/f^*(INS) \leq 2$, *where* $f^*(INS)$ *is the number of c-rectangles in an optimal solution for the instance INS of the* $1CR_R$ *problem.*

PROOF. The proof is omitted since it is similar to the one for Theorem 3. □

A simple linear time algorithm with an approximation bound of two just assigns one rectangle per square in the c-simple instance $S(X)$ obtained from X. This is not the only linear time algorithm to achieve this approximation bound. Any linear time algorithm that generates solutions in which adjacent array elements assigned the same color are tight in the same c-rectangle has the same approximation bound.

2.4 An improved algorithm for the $1RCR_R$ *and* $1RCNR_R$ *problem*

In this subsection we present an improved algorithm for the $1RCR_R$ and the $1RCNR_R$ problems. Remember that for these two problems each color is assigned to at most two of the array elements. It is simple to prove that the $1RCR_R$ and the $1RCNR_R$ are identical problems, i.e., any solution to one of these problems can be easily converted to a solution to the other problem. The dynamic programming algorithm for the $1CR_R$ reduces to an $O(n^2)$ algorithm for $1RCR_R$. We show that in general there exists a faster algorithm for this case, by reducing our problems to the problem of finding a maximum independent set in an overlap graph. Before presenting our reduction we define overlap graphs as well as other useful terms.

We refer to a horizontal line segment that overlaps with the x-coordinate axis as an *interval.* We say that two intervals *overlap* if they intersect, but neither fully contains the other. A graph is called an *overlap graph* if there is a one-to-one correspondence between the vertices of the graph and a

collection of intervals such that two vertices are adjacent if and only if their corresponding intervals overlap [5]. The problem of finding a maximum independent set in an overlap graph can be solved in $O(dn)$ time [1], where d is the *density*, i.e. the maximum number of intervals including any point.

Let us now reduce the $1RCR_R$ problem to the problem of finding a maximum independent set in an interval graph. Consider an instance for the $1RCR_R$ problem. Each color assigned to two entries in the array is represented by an interval whose extreme points are the centers of the two array entries. Let G be the overlap graph induced by the set of intervals and let S be a maximum independent set for G. A solution to the $1RCR_R$ problem from S is constructed as follows. For each vertex v in S we define the rectangle D_v which minimally covers both elements in the array colored by the color of the interval that v represents. The r-compression consists of all the D_v rectangles from largest to smallest followed by a set of 1 by 1 squares for the remaining elements in the graph. We claim that the r-compression constructed this way is an optimal solution to the $1RCR_R$ problem.

THEOREM 5. *The above procedure solves the $1RCR_R$ and the $1RCNR_R$ in $O(dn)$ time, where n is the number of entries and d is the maximum number of "intervals" that include any point.*

3. Two dimensional maps

We begin by showing that the $2RCR_R$ problem is *NP*-hard. This implies that problems $2RCNR_R$, $2CR_R$ and $2CNR_R$ are also *NP*-hard problems. With a little adjustment of the proof we show that problems $2CR_{RP}$ and $2CNR_{RP}$ are *NP*-hard problems. We also show that the problems remain *NP*-hard even when the number of different colors in M is bounded by a small constant ($2CR_R(4)$ and $2CNR_R(2)$).

3.1 The $2RCR_R$ and $2RCNR_R$ problems are NP-hard

Before we establish that the $2RCR_R$ and $2RCNR_R$ problems are *NP*-hard, we define some terms and make some observations.

DEFINITION 2. *Let D' be the set of colors that appear in exactly two entries in M. For each color $d \in D'$, let D_d be the minimal rectangle enclosing both elements colored d in M. Such rectangle is called a* minimum 2-element *rectangle.*

Note that for any instance of the $2RCR_R$ and the $2RCNR_R$ problem there exists a minimum r-compression comprised solely of minimum 2-element rectangles and 1×1 squares, otherwise we could shrink the rectangles until all of them are of one of these two types. Furthermore, we can assume that for each color $d \in D'$ there is either *one* minimum 2-element rectangle, or two 1×1 squares assigned this color in the r-compression. We refer to an r-compression of the above form as a *normal r-compression.* Let

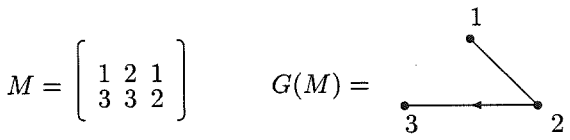

Fig. 4: Example of a matrix with its matrix graph.

$D_{d_i}(1 \leq i \leq k)$ be the minimum 2-element rectangles in a minimum normal r-compression. Then the total number of rectangles needed to cover M is $nm - k$.

DEFINITION 3. *The* matrix graph *of M, $G(M)$ is a directed graph whose vertices represent the minimum 2-element rectangles ($D_d, d \in D'$) and whose edges are directed from the vertex representing D_{d_1} to the vertex representing D_{d_2} if and only if the minimum 2-element rectangle D_{d_1} contains an element in M colored d_2.*

Since the proof of the following lemma is straightforward, it will be omitted. Example 2 further illustrates the concepts just introduced.

LEMMA 2. *The graph $G(M)$ has an induced acyclic subgraph with p vertices if and only if there exist an r-compression of M with $(nm - p)$ c-rectangles.*

EXAMPLE 2. Fig. 4 illustrates an example of a matrix M and its corresponding matrix graph $G(M)$. An undirected edge in the figure represents two way edges. The maximum acyclic subgraph contains two vertices. Thus the minimum number of rectangles needed to cover M is 4.

We show that the $2RCR_R$ problem is *NP*-hard by reducing a restricted version of the exact cover by three sets to it. We shall refer to this problem as the *RXC*3 problem. The *RXC*3 problem was shown to be *NP*-complete in [6].

DEFINITION 4. **Restricted exact cover by three sets (*RXC*3):** *Given a finite set of elements $X = \{x_1, \ldots, x_{3q}\}$ and a collection $S = \{S_t \mid 1 \leq t \leq r, S_t \subset X$ and $\mid S_t \mid = 3\}$ of $r = 3q$* S-sets *(3-element subsets of X) in which each element of X appears in exactly three S-sets and no element of X appears more than once in the same S-set. The RXC3 problem consists of determining whether or not S has an exact cover for X, i.e. a subcollection S' of S such that every element of X occurs in exactly one member of S'.*

First we transform the *RXC*3 problem into the problem of determining whether or not an undirected graph has an independent set of some given

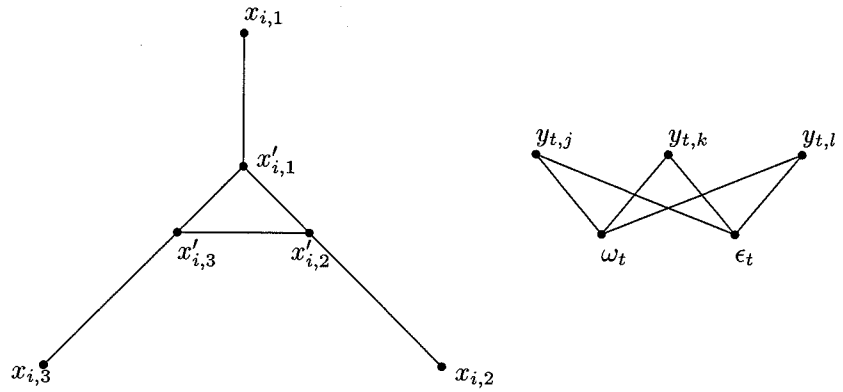

Fig. 5: X-subgraph and S-subgraph.

size. Then we transform it into the problem of determining whether a directed graph has an induced acyclic subgraph with certain number of vertices. The transformation to our map compression problem is obtained from the directed graph generated in the last reduction.

Given an instance of the *RXC*3 problem we construct an undirected graph G as follows. For each element $x_i \in X$ we introduce a subgraph on vertices $x_{i,j}$ and $x'_{i,j}$ for $j = 1, 2, 3$ as illustrated in Fig. 5 (left). We refer to these subgraphs as *X-subgraphs*. For each S-set $S_t = \{x_j, x_k, x_l\}$ we introduce a $K_{2,3}$ subgraph with vertices $y_{t,j}$, $y_{t,k}$, $y_{t,l}$, ω_t and ϵ_t (see Fig. 5 (right)). We refer to these subgraphs as *S-subgraphs*.

DEFINITION 5. *For each $x_i \in X$, let $f_1(x_i) \geq f_2(x_i) \geq f_3(x_i)$ be the indices of the S-sets that contain x_i. Since in every instance of the RXC3 problem no element in X may appear more than once in an S-set, we know that $f_1(x_i) > f_2(x_i) > f_3(x_i)$.*

The graph G has also the following edges between the X-subgraphs and the S-subgraphs. For $x_i \in X$ there is an edge between $x_{i,j}$ and $y_{f_j(x_i),i}$, for $j = 1, 2, 3$. This completes the definition of the graph G. When X has $3q$ elements, the graph G has $6 \cdot 3q + 5 \cdot r = 33q$ vertices and $6 \cdot 3q + 6r + 3 \cdot 3q = 45q$ edges. The following lemma establishes an important property of graph G.

LEMMA 3. *Any independent set of vertices in G has at most $16q$ vertices. Furthermore, G has an independent set with $16q$ vertices if and only if (X, S) has an exact cover.*

PROOF. Suppose G has an independent set L with more than $16q$ vertices. Let L_1 be the restriction of L to the X-subgraphs (i.e., the set of $x_{i,j}$ and $x'_{i,j}$ vertices in L) and L_2 be the restriction of L to the S-subgraphs. Clearly, $| L | = | L_1 | + | L_2 |$. Let α_i be the number of X-subgraphs with exactly

i vertices in L_1 $(i = 0, 1, 2, 3)$. Let β_i be the number of S-subgraphs with exactly i vertices in L_2 $(i = 0, 1, 2, 3)$. We can assume that $\alpha_0 = 0$ as otherwise the independent set is not maximal, since we can add to the independent set at least one $x'_{i,j}$ vertex in the X-subgraph. Similarly, we can assume $\beta_0 = \beta_1 = 0$. It is simple to show that the following statements hold.

$$\begin{aligned} 3q &= \alpha_1 + \alpha_2 + \alpha_3 \\ 3q &= \beta_2 + \beta_3 \\ |\, L_1 \,| &= \alpha_1 + 2\alpha_2 + 3\alpha_3 \\ |\, L_2 \,| &= 2\beta_2 + 3\beta_3 \end{aligned}$$

Since each y vertex in an S-subgraph that is in L_2 is adjacent to an $x_{i,j}$ vertex in an X-subgraph, and each X-subgraph with exactly i $(1 \leq i \leq 3)$ vertices in L_1 has at most $(4 - i)$ $x_{i,j}$ vertices that are not in L_1, we know that

$$3\beta_3 \leq 3\alpha_1 + 2\alpha_2 + \alpha_3.$$

Thus we obtain

$$|\, L_1 \,| + |\, L_2 \,| \leq 4(\alpha_1 + \alpha_2 + \alpha_3) + 2\beta_2 = 12q + 2\beta_2.$$

By assumption $|\, L \,| \geq 16q$, so we know that $\beta_2 \geq 2q$. Since $3q = \beta_2 + \beta_3$, it must be that $\beta_3 \leq q$. Thus

$$|\, L_2 \,| = 2\beta_2 + 3\beta_3 = 6q + \beta_3 \leq 7q$$

and hence $|\, L_1 \,| \geq 9q$. Since $|\, L_1 \,| = \alpha_1 + 2\alpha_2 + 3\alpha_3$ and $3q = \alpha_1 + \alpha_2 + \alpha_3$, we know that

$$|\, L_1 \,| = \alpha_1 + 2\alpha_2 + 3\alpha_3 \leq 9q.$$

So it must be that $|\, L_1 \,| = 9q$ and $|\, L_2 \,| \leq 7q$. Therefore, $|\, L \,| \leq 16q$ which contradicts our assumption. Hence, any independent set of G has at most $16q$ vertices.

Let us now show that G has an independent set with $16q$ vertices if and only if (X, S) has an exact cover. First we show that if (X, S) has an exact cover, then G has an independent set with cardinality $16q$. From any exact cover for (X, S) we construct an independent set, T, as follows. Initially T is empty. For each S-set in the exact cover we add to T the three y vertices in the corresponding S-subgraph and for each S-set not in the exact cover we add to T the ω and ϵ vertices. For each element x_i and $j \in \{1, 2, 3\}$, if $f_j(x_i)$ is an S-set in the exact cover, we add to T vertex $x'_{i,j}$, otherwise we add $x_{i,j}$. It is easy to verify that T is an independent set with $16q$ vertices.

Now we show that if G has an independent set with cardinality $16q$, then (X, S) has an exact cover. From the above discussion we know that if G has an independent set with $16q$ vertices, then $|\, L_1 \,| = 9q$ and $|\, L_2 \,| = 7q$. Also, by the above discussion we know that $\alpha_3 = 3q, \alpha_1 = \alpha_2 = 0, \beta_3 =$

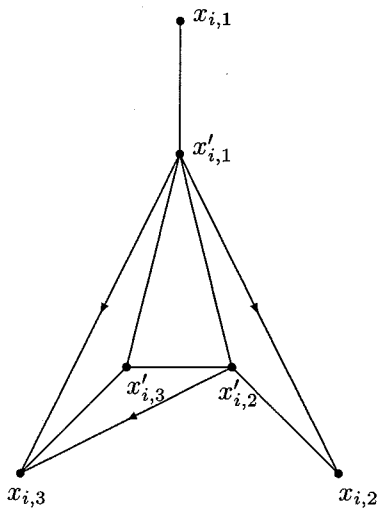

Fig. 6: X-subgraph in H.

q and $\beta_2 = 2q$. Consider the set S' of q S-sets whose corresponding S-subgraphs have three elements in the independent set of G. Now we show that S' is an exact cover for X. All the X-subgraphs have one $x'_{i,j}$ vertex and two $x_{i,j}$ vertices in the independent set, otherwise we can replace an $x_{i,j}$ vertex by an $x'_{i,j}$ in the independent set. Therefore, any element in X is in at most one S-set in S'. Now counting the edges between the S-subgraphs of S' and the X-subgraphs which have an edge to an S-set of S' we know that every element is in exactly one S-set of S'. Therefore, (X, S) has an exact cover if and only if G has an independent set with $16q$ vertices. □

Now we transform the graph G into a directed graph H as follows. The vertices of H are the same as those of G. We replace every edge of G by two way directed edges (one directed each way). Next, in the X-subgraphs we add edges from $x'_{i,1}$ to $x_{i,2}$ and $x_{i,3}$, and from $x'_{i,2}$ to $x_{i,3}$. So the X-subgraphs of H are of the form given in Fig. 6 (two way edges are drawn as undirected edges). The following lemma establishes an important property of the H graphs.

LEMMA 4. *Any induced acyclic subgraph of H has no more than $16q$ vertices. Furthermore, H has an induced acyclic subgraph with $16q$ vertices if and only if graph G has an independent set with $16q$ vertices.*

PROOF. Consider an induced acyclic subgraph in H. Clearly, these vertices induce an independent set in G. Thus the subgraph has at most $16q$ vertices and if H has an acyclic subgraph with $16q$ vertices then G has an independent set of the same cardinality. Now suppose G has an independent set with $16q$ vertices. Since the extra edges we have added to the X-subgraphs

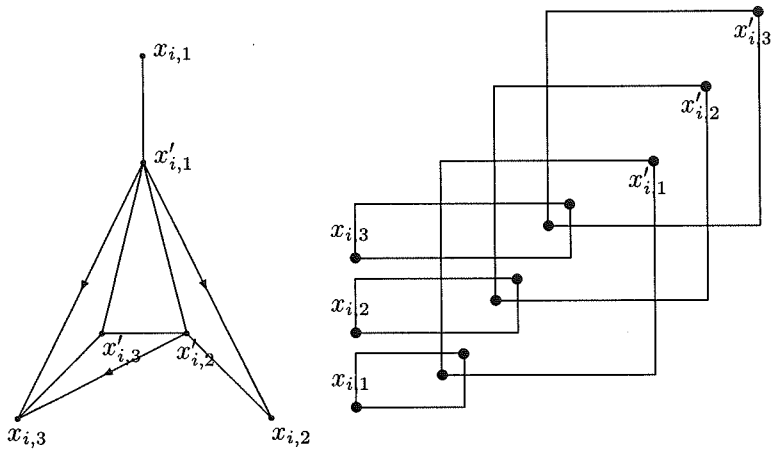

Fig. 7: X-subgraph and rectangle representation.

do not by themselves form a cycle, we have that the independent set induces an acyclic subgraph in H. □

Now we construct a matrix M such that $G(M) = H$ and the size of M is bounded by a polynomial in q (remember that $\mid X \mid = 3q$). We refer to this construction as the *rectangle representation* of H. Consider the graph H. Each vertex v of H is represented by a rectangle D_v whose two diagonal corners (bottom-left and top-right) are colored v. First consider the X-subgraphs. The rectangle representation of these subgraphs is given in Fig. 7. The solid dots in the rectangles indicate which of the rectangle corners are to be colored with the color representing the vertex in G. It is easy to verify that these rectangles correctly represent the interactions in the X-subgraph. Now consider the S-subgraphs. The rectangle representation of these subgraphs is given in Fig. 8.

Remember that vertex $x_{i,j}$ is adjacent to vertex $y_{f_j(x_i),i}$, for all $x_i \in X$ and $j = 1, 2, 3$. To account for these interactions, we simply assign the lower left corner of the rectangle for $x_{i,j}$ so that it just includes the right upper corner of $y_{f_j(x_i),i}$. This is illustrated in Fig. 9. Now we rearrange the rectangles in a matrix M in such a way that only the interactions given by H are allowed. In order to do this we first partition M as in Fig. 10. In this figure the S_i's are the rectangle representations of the S-subgraphs and the X_i's are the upper parts of the rectangle representations of the X-subgraphs. Note that in order to get the correct X-subgraph we have to be sure that $x_{i,1}$ is adjacent to the S-subgraph which appears lowest in the matrix (otherwise $x'_{i,1}$, $x'_{i,2}$ and $x'_{i,3}$ will not form a triangle, see Fig. 9). This, however, is guaranteed by choosing the functions $f_j(x_i)$ such that $f_1(x_i) > f_2(x_i) > f_3(x_i)$. From our construction it follows that the graph

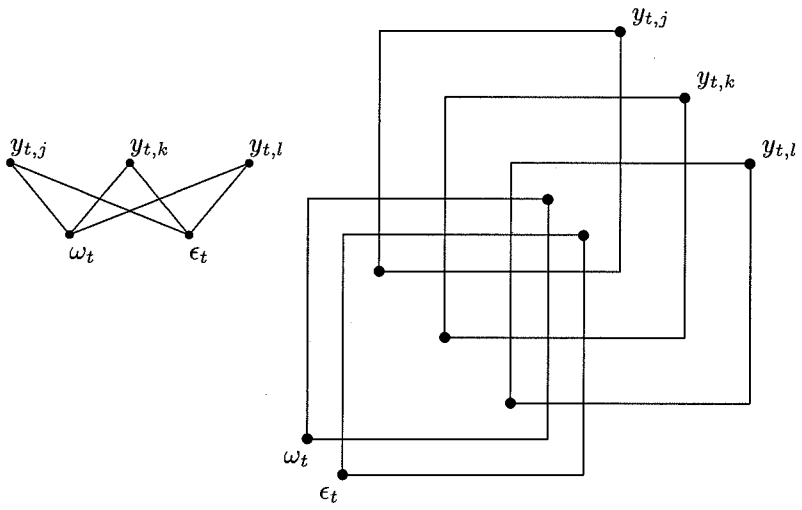

Fig. 8: S-subgraph and rectangle representation.

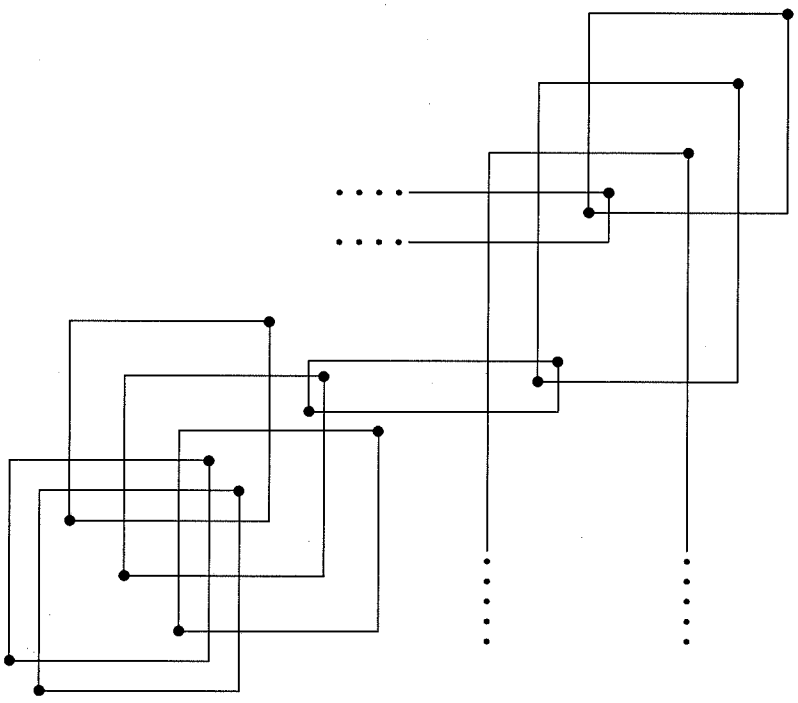

Fig. 9: Connecting $x_{i,j}$ and $y_{f_j(x_i),i}$.

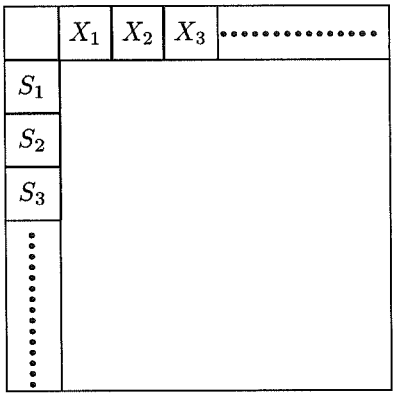

Fig. 10: Partition of M.

for the rectangles $(D_v)_{v \in H}$ equals H. The remaining matrix entries in M are filled with colors that appear only once.

It is clear that one can construct the matrix M with dimensions bounded by a polynomial in q (the input size of $RXC3$ is $O(q)$). By Lemma 2, Theorem 6 and Lemma 3 this completes the proof that the $2RCR_R$ problem is NP-hard. This result and obvious extensions are stated in the following theorem.

THEOREM 6. *The $2RCR_R$, $2RCNR_R$, $2CR_R$ and $2CNR_R$ problems are NP-hard.*

PROOF. By the above discussion. □

3.2 The $2CR_{RP}$ and $2CNR_{RP}$ problems are NP-hard

In this section we consider the $2CR_{RP}$ and $2CNR_{RP}$ problems. We show that the $2CR_{RP}$ problem is NP-hard by reducing the $RXC3$ problem to it. As a starting point we take the partly filled matrix M constructed from the rectangle representation of the graph H in the reduction for the $2RCR_R$ problem. We may assume that two occurrences of the same color do not occur in the same row or column of M. We proceed to replace every element M_{ij} by a 2×2 submatrix S_{ij}. Consider the two occurrences of a color c in M say (i,j) and (k,l). Without loss of generality assume $i < k$. The two squares in S_{ij} and S_{kl} farthest from each other are colored c. The other three squares in $S_{i,j}$ are colored $\bar{c}$. The three occurrences of $\bar{c}$ are contained in D_c, (the minimal rectangle which encloses both elements c). We call the matrix constructed in this way A. The remaining entries in A is filled with colors that are used only once.

We have three types of colors in A. One type corresponds to vertices of the graph H. We refer to these colors as v-colors. Another type are the

colors that appear three times in A. These are L-colors. The last type of colors are those that appear only once and we call these d-colors. Let d be the number of d-colors in A and let n be the number of vertices of H (so there are n v-colors in A).

LEMMA 5. *The graph H has a maximum induced acyclic subgraph of p vertices if and only if there is an rp-compression of A such that the total number of corners is $12n - 2p + 4d$.*

PROOF. Suppose H has a maximum induced acyclic subgraph S of p vertices. We construct an rp-compression of A as follows.

(1) Find a sequence of rectangles covering the v-colors representing the vertices of S. There are p of these rectangles and the total number of corners is $4p$.

(2) The L-colors inside these rectangles are covered by L-shaped RP's, with 6 corners each. There are p of these RP's and the total number of corners is $6p$.

(3) The other L-colors are covered by 2×2 squares. There are $n - p$ such squares and the total number of corners is $4(n - p)$.

(4) The rest of the v-colors are covered by two 1×1 squares. This gives $2(n - p)$ such squares and the total number of corners is $8(n - p)$.

(5) Each d-color is covered by a 1×1 rectangle. There are d such rectangles and the total number of corners is $4d$.

The rp-compression has at total of $12n - 2p + 4d$ corners.

Now consider an rp-compression of A with $12n - 2p + 4d$ corners. We first show that we can assume that all v-colors are covered by rectangles. Consider the RP's covering a v-color a. If the number of corners of these RP's is at least 8 then we can replace these RP's with two 1×1 squares. The only other case is when a is covered by an L-shaped RP with 6 corners. Now consider the L-color inside D_a, say a'. Clearly, the RP's covering a' must have at least 6 corners (one L-shape rectilinear polygon). Thus in total we have at least 12 corners to cover a and a'. Replace the L-shape covering a by two 1×1 squares, and the RP's covering a' by one 2×2 square. This new covering of a and a' by rectilinear polygons has a total of 12 corners. Thus we can assume that all v-colors are covered by rectangles. Of course we can also assume that the rectangles are minimal, i.e. a v-color a is either covered by D_a or by two 1×1 squares. Now consider the v-colors of H which are covered by *one* rectangle. Clearly, these vertices do not contain a cycle in H. It follows that the number of these colors is p. □

This lemma has essentially established the NP-hardness of the $2CR_{RP}$ problem. Note that for the matrix A we have constructed the problems $2CR_{RP}$ and $2CNR_{RP}$ are identical. Hence we have the following theorem.

THEOREM 7. *The $2CR_{RP}$ problem and the $2CNR_{RP}$ problem are NP-hard.*

PROOF. By the above discussion. □

4. Maps with a constant number of colors

In the previous section we showed that the $2CR_R$, $2CNR_R$, $2CR_{RP}$ and $2CNR_{RP}$ problems are *NP*-hard. However, in many practical cases the number of colors is small compared to the size of the matrix. So the question remains whether a restriction on the number of colors makes the problem computationally tractable. In this section we show that even the $2CR_R(4)$ and the $2CNR_R(2)$ problems are *NP*-hard problems. In addition, we provide evidence that the $2CNR_R(2)$ problem is hard to approximate.

4.1 NP-hardness of the $2CR_R(4)$ problem

In this section we show that the $2CR_R(4)$ problem is *NP*-hard by reducing the feedback vertex set to it. First we define the feedback vertex set problem, formulate related problems and prove some useful properties about them. Then we present a polynomial transformation and prove a useful property about it. The main theorem, whose proof is based on these results, is given at the end of the section. The feedback vertex set problem, formally defined below, is *NP*-complete [4].

DEFINITION 6. **Feedback vertex set.**
Instance: A directed graph $H = (V_H, E_H)$, and integer $l \leq \mid V_H \mid$.
Question: Is there a set $W \subseteq V_H$ such that $H[V_H - W]$ (i.e., the graph induced by the set of vertices in $V_H - W$) is acyclic and $\mid W \mid \leq l$?

Let H be any graph and an l be an integer. The graph $G = (V, E)$ is defined as the disjoint union of two copies of H, and let $k = 2l$. Obviously, G has a feedback vertex set of size k, if and only if H has a feedback vertex set of size l. Furthermore, the minimum size of a feedback vertex set in G is even.

From G we construct the directed graph $G' = (V_1 \cup V_2, E')$, where $V_i = \{v_j^i \mid v_j \in V\}$ for $i = 1, 2$, and $E' = \{(v_j^1, v_j^2) \mid v_j \in V\} \cup \{(v_j^2, v_k^1) \mid (v_j, v_k) \in E\}$. Obviously, G' is bipartite, and $(v_j^2, v_k^1) \in E' \Rightarrow (v_k^1, v_j^2) \notin E'$. Let $n = |V_1| = |V_2|$.

LEMMA 6. *The following statements are equivalent:*

(1) G has a feedback vertex set W of size at most k (i.e. a set $W \subseteq V$, $\mid W \mid \leq k$ and $G[V - W]$ acyclic).

(2) G' has a feedback edge set F of size at most k (i.e. a set $F \subseteq E'$ such that $\mid F \mid \leq k$ and directed graph $(V_1 \cup V_2, E' - F)$ is acyclic).

(3) G' has a feedback vertex set W' of size at most k.

PROOF. (1) $\Rightarrow$ (2): Let $F = \{(v_j^1, v_j^2) \mid v_j \in W\}$. (2) $\Rightarrow$ (3): Let $W' = \{v_i^1 \mid \exists_{v_j^2}[(v_i^1, v_j^2) \in F \text{ or } (v_j^2, v_i^1) \in F]\}$. (3) $\Rightarrow$ (1): Let $W = \{v_i \mid \exists_i[v_i^1 \in W' \text{ or } v_i^2 \in W']\}$. □

Before we define the matrix M for the $2CR_R(4)$ problem that we construct from G', we introduce a set of objects. We define a *horizontal (vertical) strip* as a set of adjacent matrix entries located along the same row (column), and a *square* as a submatrix of adjacent rows and columns with the same number of rows and columns. The first object is a white square that contains all the matrix elements in M. For each vertex $v_i^1 \in V_1$ we define a red horizontal strip, and for each vertex $v_i^2 \in V_2$ we define a green vertical strip. These objects are referred to as the horizontal and vertical vertex-strips, respectively. The area of a strip that represents vertex $v \in V_1 \cup V_2$ is denoted by $A(v)$. There are $4L(n+1)$ objects called *extra-strips*, where $L = 8k + 9$. Half of these extra-strips are red horizontal strips, and $2L$ of them are located above the topmost horizontal vertex-strip, in between each pair of adjacent horizontal vertex-strips, and below the bottommost horizontal vertex-strip. Similarly, there are $2L$ vertical black extra-strips located to the left of the leftmost vertical vertex-strip, in between each pair of adjacent vertical vertex-strips, and to the right of the rightmost vertical vertex-strip. Each horizontal strip intersects all the vertical strips. Between every pair of horizontal (vertical) strips there is at least one row (column) without a horizontal (vertical) strip. Further, the leftmost and rightmost (topmost and bottommost) matrix entry of any two horizontal (vertical) strips are located on different columns (rows).

For vertices v_i^1, v_j^2, if $(v_i^1, v_j^2) \notin E$ and $(v_j^2, v_i^1) \notin E$, define a black square of size $2i + 2j - 1$ centered at the matrix entry in $A(v_i^1) \cup A(v_j^2)$. There is enough space between the horizontal (vertical) strips so that the black squares do not intersect nor are neighbors of the extra-strips.

Partition each of the sets with $2L$ consecutive extra-strips (i.e., the ones between consecutive vertex-strips) into two sets of L consecutive extra-strips. Let S_1 and S_2 be the resulting sets.

Let us now define from these objects the matrix M. The first rule that applies to a matrix entry defines its color.

(1) The matrix entries in a black square are colored black.

(2) The matrix entry $A(v_i^1) \cup A(v_j^2)$ is colored green when $(v_i^1, v_j^2) \in E$.

(3) The matrix entry in $A(v_i^1) \cup A(v_j^2)$ is colored red when $(v_j^2, v_i^1) \in E$. Note that by construction (G') it cannot be that case 2 and 3 apply to the same matrix entry.

(4) Color red each matrix entry in $A(v_i^1)$; and color green each matrix entry in $A(v_j^2)$.

(5) Color red each matrix entry in a horizontal extra-strip in S_2.

(6) Color black each matrix entry in a vertical extra-strip in S_2.

(7) Color red each matrix entry in a horizontal extra-strip in S_1.

(8) Color black each matrix entry in a vertical extra-strip in S_1.

(9) Color white each matrix entry in the white square.

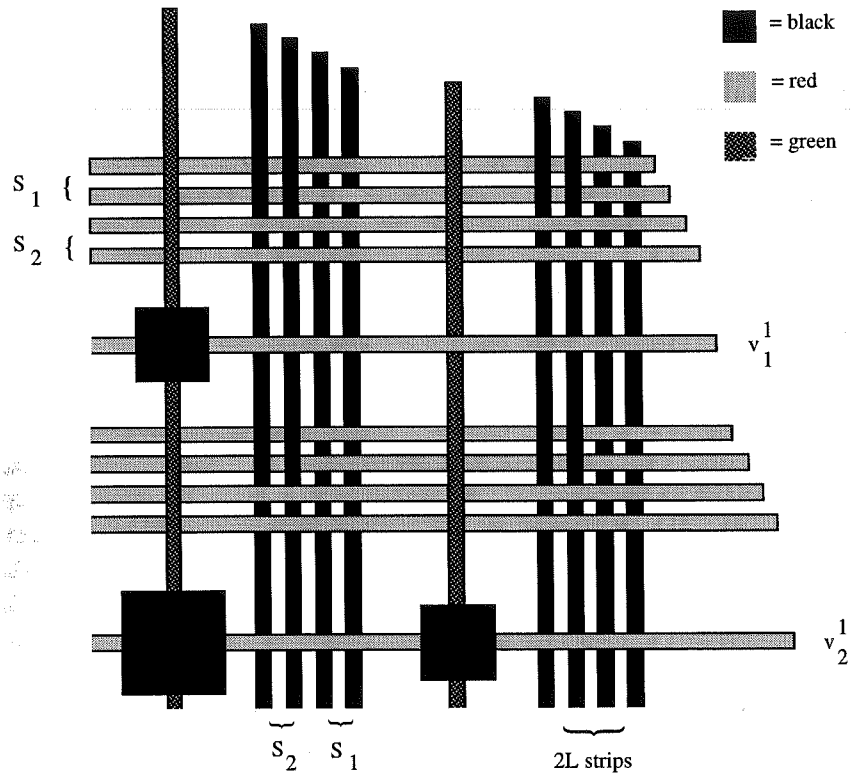

Fig. 11: Example of map constructed.

The construction is illustrated with an example in Fig. 11 (only the right upper part of the figure is shown.) Note that the only objects that preserve their colors in M are the black squares. Let $B = n^2 - |E'|$ be the number of black squares (remember that these are not 1 by 1 squares). In the following lemma we establish the basic property of our transformation.

LEMMA 7. *G has a feedback vertex set of size at most k if and only if there is an r-compression for the matrix M with at most $B+2n+1+4L(n+1)+k$ c-rectangles.*

PROOF. $\Rightarrow$: By Lemma 6 we know that if G has a feedback vertex set of cardinality at most k then G' has a feedback edge set of size at most k. Let F be a feedback edge set of G' of size at most k. Let $x_1, x_2, \ldots x_{2n}$ be a topological sort of the vertices in the graph $G' - F = (V_1 \cup V_2, E' - F)$. Let us now construct an r-compression for the matrix M with at most $B + 2n + 1 + 4L(n+1) + k$ c-rectangles. The first c-rectangle is the white square that covers all the matrix elements. This is followed by a sequence of c-rectangles that correspond to each of the vertical black extra-strips in S_1, then to each of the horizontal red extra-strips in S_1, then to each of the vertical black extra-strips in S_2, and then to each of the horizontal red

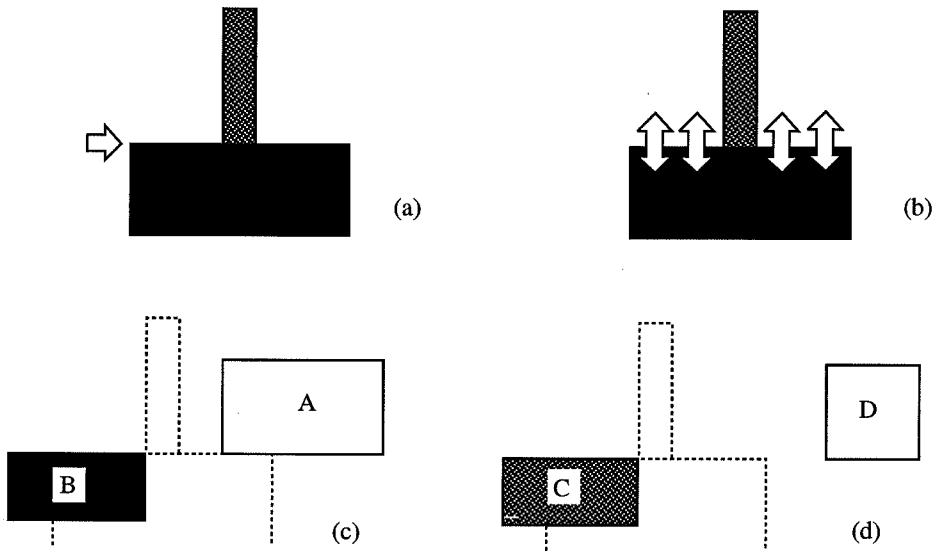

Fig. 12: Example of tokens. (a) The upper side of a black square. (b) Pairs of pixels, corresponding to the token representing this upper side. (c) Rectangles A and B satisfy the token. (d) Rectangles C and D weakly satisfy the token.

extra-strips in S_2. The next $2n$ c-rectangles cover exactly the matrix entries for the vertex-strip for $A(x_1)$, then $A(x_2)$, and so on. These c-rectangles are followed by a sequence of B c-rectangles each corresponding to one of the black squares. At this point the only matrix elements that have not been assigned their correct color are matrix elements located at the intersection of the red horizontal vertex-strip for vertex $x_i \in V_1$ with a green vertical vertex-strip for the vertex $x_j \in V_2$, and either $i < j$ and $(x_j, x_i) \in E$ (that is the position is green but must be red) or $j < i$ and $(x_i, x_j) \in E$ (then the position is red but must be green). In both cases the edge (either (x_i, x_j) or (x_j, x_i)) belongs to the feedback edge set F. So $\mid F \mid \leq k$ matrix elements have not been assigned the correct color. This can be remedied with $\mid F \mid$ c-rectangles of size 1 by 1. Therefore, there is an r-compression with at most $B + 2n + 1 + 4L(n+1) + k$ c-rectangles.

$\Leftarrow$: Assume that there exists an r-compression with at most $B + 2n + 1 + 4L(n+1) + k$ c-rectangles. From Lemma 6 we know that to complete the proof of the lemma we only need to show that there exists a feedback edge set of G' of size at most k.

To establish this fact we make extensive use of *tokens*. A *token* represents a collection of pairs of adjacent matrix entries. All the first elements of the pairs in a token have the same color in M, and all the second elements of the pairs in a token have the same color in M, but the two colors are different. There are two types of tokens: horizontal and vertical. All first elements of pairs in a horizontal (vertical) token lie in the same row (column), and all the second elements lie in the row (column) below (to its right).

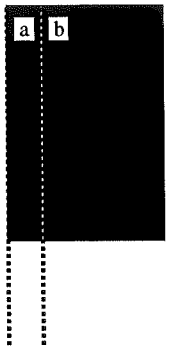

Fig. 13: A c-rectangle that satisfies the upper side of a vertical strip and also colors pixel (b) black.

We use T_H (T_V) to represent the following set of horizontal (vertical) tokens. Each strip (square) has two tokens in T_H and two tokens in T_V, each corresponding to a side of the strip (square), except for the white square. The colors in these tokens are always chosen to be the unordered pair { white, color of the strip or square }.

For each matrix entry pair in each token there must be at least one c-rectangle in any r-compression that contains exactly one of the two matrix entries and the color of the c-rectangle is the same color as the matrix entry it includes. Any such c-rectangle is said to *satisfy* the token. A c-rectangle is said to *weakly satisfy* a horizontal (vertical) token, if it satisfies the token, or it has matrix entries located on exactly one of the two rows (columns) where the matrix entries of the pairs in the tokens are located. In other words, a c-rectangle weakly satisfies a horizontal token, if it would satisfy the token after a horizontal shift and/or a color change. For an example, see Fig. 12.

Note that each token in T_H and in T_V must be satisfied by at least one c-rectangle in an r-compression. We say that a (side of) a c-rectangle satisfies a side of a strip, if it satisfies the corresponding token. It is important to note that we have defined all strips and squares in such a way that all the tokens in T_H and T_V are different, and each c-rectangle can (weakly) satisfy at most two horizontal and two vertical tokens. This property is exploited throughout our proof. It is simple to show that there are $2B+4n+8L(n+1)$ tokens in T_H, and in T_V.

From the above observations it is simple to see that an r-compression satisfies all tokens only if it has at least $B+2n+4L(n+1)$ c-rectangles. To this number, we should add the number of times a side of c-rectangle weakly satisfies a token that has already been satisfied, or does not satisfy a token at all, divided by four. So, we may assume that in our r-compression at most $4(k+1)$ sides of c-rectangles weakly satisfy an already satisfied token or satisfy no token at all.

Let us now determine the order in which the c-rectangles that color the extra-strips appear in the r-compression. An extra-strip is said to be *well behaved* in an r-compression, if there is a c-rectangle in the compression equal to the strip (same area and color), none of the tokens associated with the strip are satisfied more than once, and the tokens corresponding to the long sides of the strip are not weakly satisfied more than once. In other words, an extra-strip is not well behaved, if there is no c-rectangle with area and color exactly equal to this strip, or at least one of its sides is satisfied more than once, or at least one of its long sides is weakly satisfied more than once.

We claim that the number of strips that are not well behaved is at most $8(k+1)$. First we establish that for each extra-strip that is not well behaved, either one of the tokens corresponding to a side of the extra-strip is satisfied or weakly satisfied at least twice, or there is at least one side of a c-rectangle that can be associated with with the extra-strip and possibly associated to another extra-strip that does not satisfy any token (the associated is between a side of a c-rectangle and extra-strips). There is nothing to prove when the the extra-strip is not well behaved and one of its tokens is satisfied or weakly satisfied at least twice. Let us consider the other case. Suppose that the tokens of the extra-strip are satisfied or weakly satisfied only once and there is no c-rectangle in the r-compression equal to the extra-strip. We now associate at least one side of a c-rectangle that does not satisfy or weakly satisfy a token with this extra-strip. This side of the c-rectangle may be associated with at most two extra-strips. We only discuss the case when the extra-strip is a vertical strip, since the proof for the other case is similar. Consider the c-rectangle that satisfies the upper small end of the extra-strip. Since it is satisfied or weakly satisfied only once, we know that the c-rectangle must be black. If it has width one, then we know that the other small end does not satisfy any token, since the c-rectangle does not cover exactly the extra-strip. Otherwise (the width is greater than one), a situation like in Fig. 13 arises. Matrix entry (b) is colored black just after the black c-rectangle appears in the r-compression, but must end up white. If it is colored white by a c-rectangle that also contains (a), then the upper side of the extra-strip has to be satisfied a second time, which by assumption does not occur. Therefore, the upper side of the c-rectangle R that colors (b) white does not satisfy a token. This upper side of R is associated with the extra-strip. Since the same situation can occur on the other side of R, the same side of R may be associated with another extra-strip. It is simple to see that such side is never associated more than twice.

From the above discussion we know that for each extra-strip that is not well behaved we can add at least $1/8$ of an additional c-rectangle to a number of $B+2n+4L(n+1)$. Since there are no more than $B+2n+1+4L(n+1)+k$ c-rectangles in the compression, it must be that there are at most $8(k+1)$ not well behaved strips. Therefore, each collection of $2L$ successive extra-strips contains at least one well behaved strip in S_1, and one well behaved strip in S_2. It is important to note that when an extra-strip is well behaved all matrix elements adjacent to a visible matrix entry (by visible matrix

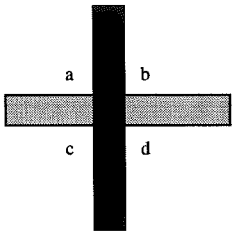

Fig. 14: The horizontal strip is drawn before the vertical strip.

entry, we mean that the color of the matrix entry is the same as the one of the extra-strip) in this extra-strip must have the correct color when the c-rectangle that color exactly the extra-strip appears in the r-compression and such entries may not be colored by a c-rectangle afterwards. This is because a token corresponding to a side of the extra-strip would be satisfied for a second time, which contradicts the definition of a well behaved extra-strip.

Next, we consider the order in which the c-rectangles for the well behaved extra-strips appear in the r-compression. Consider a well behaved horizontal extra-strip $s \in S_1$, and a well behaved vertical extra-strip $s' \in S_1$. By construction the matrix entry that lies on both strips is colored red. In case the c-rectangles that covers exactly s' appears in the r-compression after the c-rectangle that covers exactly s, then the matrix entry is black after the c-rectangle that covers s' exactly (see Fig. 14). If it is recolored by a c-rectangle of width 1, then a side of the strips is weakly satisfied for a second time. Otherwise, at least one of the four matrix entries labeled a, b, c, and d in Fig. 14 is no longer white, which also contradicts the well behavedness of the strips. Hence, we may assume that the c-rectangles for all the well behaved vertical extra-strips in S_1 appear before the c-rectangles of all the well behaved horizontal extra-strips in S_1 in the r-compression. The same argument one can be used to show that the c-rectangles of all well behaved horizontal extra-strips in S_1 appear before all the c-rectangles for all the well behaved vertical extra-strips in S_2, and the c-rectangles for all the well behaved vertical extra-strips in S_2 appear before all the c-rectangles for all the well behaved horizontal extra-strips in S_2.

Given an r-compression for M constructed from G, we partition the set of vertices in V_1 (V_2) into sets X_1 and Y_1 (X_2 and Y_2) such that $X_1 \cup X_2$ is a feedback vertex set for G. First we define a partition for V_1 and V_2, and then we show that $X_1 \cup X_2$ is a feedback set for G.

Let us now partition the set of vertices in V_1 into X_1 and Y_1. Consider the horizontal red strip corresponding to $v_i^1 \in V_1$. Vertex v_i^1 is added to set X_1 if there are at least two c-rectangles that either weakly satisfy and have the color red, or satisfy the tokens corresponding to the upper and lower sides of $A(v_i^1)$. Otherwise, vertex v_i^1 is added to set Y_1.

We claim that for each vertex v_i^1 added to set Y_1 there is a red c-rectangle with height one that exactly covers it, and in what follows we use $g(v_i^1)$ to

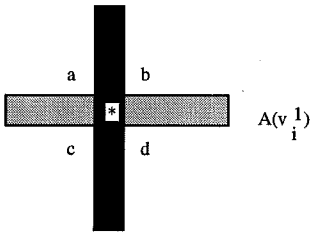

Fig. 15: Case 1 in proof.

represent such c-rectangle. By definition a vertex v_i^1 is added to set Y_1 if at least one of the tokens in its upper or lower side is satisfied only once by a c-rectangle R and it is not weakly satisfied by another red rectangle. Therefore, if R is red it must contain every matrix entry in the strip $A(v_i^1)$, or if it is white it must contain every matrix entry just above or just below the strip. We claim that R must be red and have height equal to one. Suppose this is not the case. There are two different subcases.

Case 1. The c-rectangle R is white. Consider a well behaved vertical black extra-strip s. Since parts of s visible in M are contained in R, it must be that the c-rectangle that colors exactly the vertical black extra-strip must appear after R in the r-compression. Just after such c-rectangle appears in the r-compression, we have the situation depicted in Fig. 15. The matrix entry marked (*) must end up colored red. This is impossible without either coloring (a), (b), (c), or (d), (which cannot occur due to well-behavedness), weakly satisfying a side of s, or satisfying both sides of $A(v_i^1)$. A contradiction. So it must be that R is not white.

Case 2. Suppose that R is red, and its height at least 2. (I.e., it does not satisfy the other side of the strip). A proof similar to the one in case 1 also leads to a contradiction.

Next, we partition the set of vertices in V_2 into sets X_2 and Y_2. Consider the green strip $A(v_i^2)$, and consider its intersection with the topmost and the bottommost well behaved red horizontal extra-strip in S_2. For each of these two matrix entries we know that there is at least one green c-rectangle that recolors the matrix entry after the respective red c-rectangles for the extra-strip appear in the r-compression. This also holds true for all the other well behaved extra-strips in S_1. These green c-rectangle(s) may not contain visible area of the already colored well behaved extra-strips, hence they cannot intersect a strip in S_1. Therefore, such c-rectangles cannot intersect another green strip representing a vertex in V_2. So, the only horizontal tokens these c-rectangles can satisfy are the tokens corresponding to the upper and the lower side of $A(v_i^2)$.

Count the number of green c-rectangles that appear after the last c-rectangle that colors a well behaved extra-strip in S_1, and that intersect $A(v_i^2)$ and no other $A(v_{i'}^2)$ $(i' \neq i)$. There are two cases:

Case 1. There are at least two such c-rectangles. Note that in this case these c-rectangles satisfy together at most two horizontal tokens. Put v_i^2 in set X_2.

Case 2. There is only one such c-rectangle R. It follows that R contains all the matrix entries that are in the intersection of $A(v_i^2)$ and a red strip that represents a vertex in V_1. Put v_i^2 in set Y_2, and let $g(v_i^2) = R$.

We claim that $X_1 \cup X_2$ is a feedback vertex set, or in other words, that $G[Y_1 \cup Y_2]$ is acyclic. Define the order function $f(v)$ over the set of vertices $v \in Y_1 \cup Y_2$ that gives the order in which the c-rectangles $g(v)$ appear in the r-compression, i.e., $f(v) < f(w)$, if and only if c-rectangle $g(v)$ appears before the c-rectangle $g(w)$.

Consider an edge $(v_i^1, v_j^2) \in E'$, $v_i^1 \in Y_1$, $v_j^2 \in Y_2$, and assume that $f(v_i^1) > f(v_j^2)$. Recall that the matrix entry in $A(v_i^1) \cap A(v_j^2)$ is green in M. After the c-rectangle $g(v_i^1)$ in the r-compression, this matrix entry is red. Hence, there must be a green c-rectangle R' that appears after $g(v_i^1)$, and intersects $A(v_i^1)$ and $A(v_j^2)$. If R' does not intersect another green vertical strip $A(v_{j'}^2)$, $(j \neq j')$, then $v_j^2 \in Y_1$, contradiction. Hence, R' must intersect at least two different green vertical strips. Now after the c-rectangle R' appears in the r-compression a part of $A(v_i^1)$ is colored green. Some matrix entries in this part belong to the tokens representing the long sides of $A(v_i^1)$. It follows that after the c-rectangle R' there must be another c-rectangle that satisfies a token representing a side of $A(v_i^2)$ for a second time, contradicting the definition of Y_1.

Next consider an edge $(v_j^2, v_i^1) \in E'$, $v_i^1 \in Y_1$, $v_j^2 \in Y_2$, and assume that $f(v_i^1) < f(v_j^2)$. Recall that the matrix entry α in $A(v_i^1) \cap A(v_j^2)$ is red in M. After the c-rectangle $g(v_j^2)$ appears in the r-compression this matrix entry is green. The red c-rectangle R' recoloring α must contain the matrix entry above or below α, otherwise both sides of $A(v_i^1)$ are weakly satisfied by this c-rectangle. The green c-rectangle R'' that recolors this matrix entry above or below α appears in the r-compression after all the c-rectangles for the well behaved strips in S_1; and hence, may not intersect any vertical strip in S_1, and it does not intersect with an $A(v_{j'}^2)$ for $j' \neq j$. But $g(v_j^2)$ is another c-rectangle in the r-compression as R'', so we contradict the definition of Y_2. Hence, $X_1 \cup X_2$ is a feedback vertex set for G.

Finally, we claim that the total number of c-rectangles is at least $B + 2n + 4L(n+1) + |X_1 \cup X_2|$. This follows from the facts that each horizontal side of a c-rectangle can satisfy at most one token, that there are $2B + 4n + 8L(n+1)$ tokens, at that with each vertex in $X_1 \cup X_2$ one can associate uniquely two sides of c-rectangles that do not satisfy a token, or satisfy a token that is already satisfied. Therefore, $X_1 \cup X_2$ is a feedback vertex set of size at most $k+1$. From Lemma 6, the construction of G from H, and the fact that k is even, we know that this implies that $X_1 \cup X_2$ is a feedback vertex set of size at most k. This completes the proof of the lemma. □

THEOREM 8. *The $2CR_R(4)$ problem is NP-hard.*

PROOF. We use the above transformation from the feedback vertex set problem. From the lemma and the fact that the matrix can be constructed in time polynomial in the size of G, (and hence of H), NP-hardness of $2CR_R$ follows even when restricted to four colors. □

4.2 The $2CNR_R(2)$ problem is an NP-hard

We show that the $2CNR_R(2)$ problem is an NP-hard problem by reducing the problem of covering a rectilinear polygon with at most k rectangles (CRP) to it. The CRP problem was shown to be an NP-hard problem in [2]. First we define the CRP problem and then we present our polynomial time reduction.

DEFINITION 7. **Covering a rectilinear polygon (CRP):** *Given a rectilinear polygon RP whose corners are at integer points and an integer $k \geq 3$, the problem consists of determining whether there is a set of k rectangles that cover RP without covering any point outside RP.*

THEOREM 9. *The $2CNR_R(2)$ problem is NP-hard.*

PROOF. We prove this theorem by showing that the NP-complete problem, CRP, polynomially reduces to the $2CNR_R(2)$ problem. Given any instance of the CRP problem, denoted by CR, we construct an instance of the $2CNR_R$ problem, NR. The two color (black represented by shaded areas and white represented by blank areas) matrix M for NR is given in Fig. 16, where the rectilinear polygon for the CRP problem is on the left side of the figure and a square with k^2 squares inside it is on the right side. We assume that $k \leq 2h$, where h is the number of corners in the rectilinear polygon, as otherwise the instance of the CRP problem has an answer yes [10], in which case we can construct a small instance with an answer yes. Therefore the construction of the instance takes polynomial time with respect to the size of the CR instance.

We claim that NR has an r-nr-compression with no more than $w = 3k+3$ c-rectangles if and only if CR has a covering with k rectangles. First lets prove that if CR has a covering with k rectangles, then NR has an r-nr-compression with at most w c-rectangles. We construct an r-nr-compression with w c-rectangles as follows. The first c-rectangle is colored white and covers the whole map. Then, $2k+2$ c-rectangles colored black are introduced to color the region inside the large square in the NR instance and from the k rectangle solution to the CR problem instance we define k c-rectangles to cover the rectilinear polygon in the NR instance. It is simple to show that this set of $3k + 3$ c-rectangles is an r-nr-compression for CR.

Let us now prove that if NR has an r-nr-compression with at most w c-rectangles then CR has a rectangle cover with cardinality k. Let R be any r-nr-compression for CR with at most w c-rectangles. If the c-rectangles with color black appear before all the ones with color white, then we need

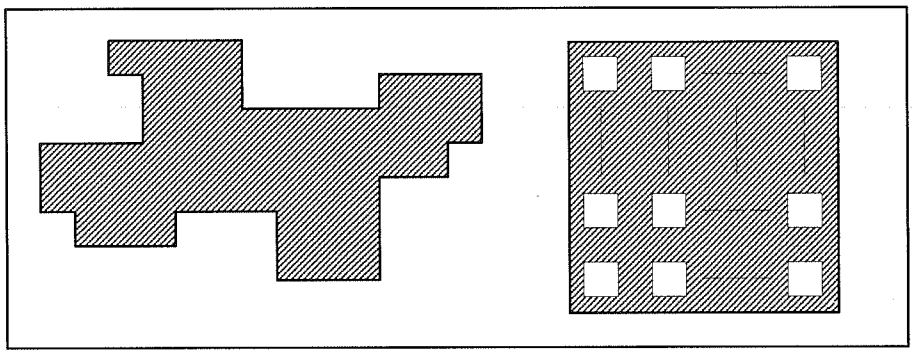

Fig. 16: Example.

at least $k^2 + 5$ c-rectangles. This is greater than w when $k \geq 3$. So it must be that the c-rectangles with color white appear before all the ones with color black. The first c-rectangle covers the entire map and it is colored white. To color the black region in the large square in NR, we need at least $2k + 2$ c-rectangles. Therefore, at most k c-rectangles are used to color the rectilinear polygon in NR. These k c-rectangles form a k rectangle cover for CR.

Hence, NR has an r-nr-compression with no more than $w = 3k + 3$ c-rectangles if and only if CR has a covering with k rectangles. This completes the proof of the theorem. □

From this reduction and the approximation algorithm for the optimization version of the CRP problem given in [10], one may conjecture that there is also an efficient approximation algorithm for the $2CNR_R(2)$ problem. Let us now present evidence to the contrary.

DEFINITION 8. **Covering a rectilinear polygon with holes ($CRP-WH$)**: *The CRP–WH problem is the same as the CRP problem, except that now the rectilinear polygon has holes (each hole is a rectilinear polygon). The problem consists of covering with k rectangles the interior of the rectilinear polygon that is not covered by holes.*

Since each instance of the CRP problem is an instance of the $CRP-WH$ problem, and the the CRP problem is an NP-complete problem, it follows that the $CRP-WH$ problem is also an NP-complete problem. In what follows when we refer to the $CRP-WH$ problem we mean its corresponding approximation problem. So far, research on developing an efficient approximation algorithm with a constant approximation bound for the $CRP-WH$ problem has been fruitless [3]. What we claim is that if there is an efficient approximation algorithm for the $2CNR_R(2)$ problem with approximation bound c', where c' is any constant, then there is an efficient approximation algorithm for the $CRP-WH$ problem with an approximation bound equal to c'', where c'' is a constant.

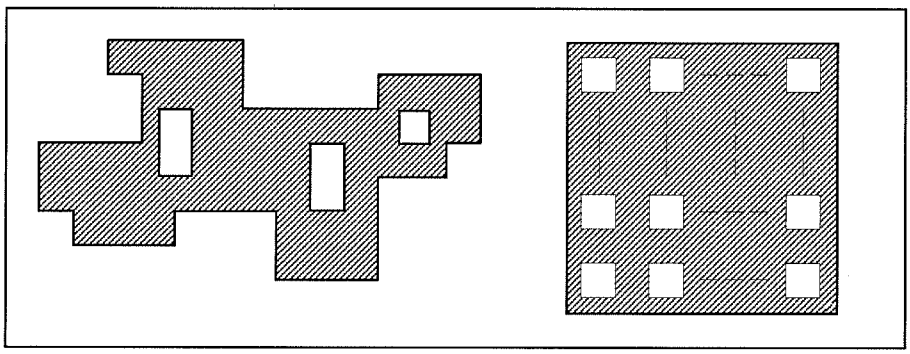

Fig. 17: Example.

THEOREM 10. *The $2CNR_R(2)$ approximation problem is as difficult as the $CRP-WH$ approximation problem, i.e., if there is an efficient approximation algorithm for the $2CNR_R(2)$ with an approximation bound equal to c', where c' is any constant, then there is an efficient approximation algorithm for the $CRP-WH$ problem with approximation bound c'', where c'' is some constant.*

PROOF. Consider any instance the $CRP-WH$ problem and let k be the number of rectangles in an optimal cover. From such instance we construct an instance of the $2CNR_R(2)$ problem as shown in Fig. 17. Clearly, an optimal solution for it has $3k+2$ c-rectangles. Now we run the approximation algorithm with approximation bound c' for the $2CNR_R(2)$ problem. We claim that all the approximations within c' of optimal are n-nr-compressions in which the first c-rectangle is white. Therefore, from such approximation we can generate a good approximation solution to the $CRP-WH$ problem. The only problem with this reduction is that we need to know the value of k in advance, which we do not know how to compute in polynomial time. In order to circumvent this problem we guess the value of k. We try $k = 1, 2, ..., h^4$, where h is the total number of boundary and hole sides in the $CRP-WH$ problem. It is simple to show that k is at most h^4. The approximation algorithm now selects the best of the solutions generated. The remaining part of the proof is omitted since the details are straightforward. □

5. Discussion

We defined a class of languages (called rectilinear) to describe digitized maps and classified them based on their level of succinct representation. For one dimensional maps, we showed that a shortest description can be generated quickly for some languages, but for other languages the problem is *NP*-hard. We also showed that a large number of linear time algorithms for our languages generate map descriptions whose length is at most twice the length

of the minimum length description. For all our languages we showed that the two dimensional map compression problem is *NP*-hard. Furthermore, for one of the most succinct of our language we presented evidence that suggests that finding a near-optimal map compression is as difficult as finding an optimal compression.

The map compression problem in which the c-rectilinear polygons must be pairwise disjoint and there is only a fixed number of different colors can be solved in polynomial time. The algorithm is based on the algorithm given in [8] and [9]. Unfortunately, such language is not a succinct one.

There are several interesting problems that remain open. The most obvious, is to develop an efficient approximation algorithm for the $2CR_R$, since this involves the most succinct of our languages. Another intersting problem, is to define other languages that are more succinct than the previous ones and for which we can develop efficient exact or approximation algorithms. Perhaps, languages based on primitive objects other than rectangles and rectilinear polygons should be investigated. For example, if the primitive objects are triangles, the resulting languages are more succinct than the rectangular ones. However, if the primitive objects are squares, the resulting languages are not as succinct as the rectangular ones. Other primitives are ellipses, simple polygons and circles. Our *NP*-hard proofs can be adapted to show that these versions of our problems in which the primitive objects are triangles and ellipses are also *NP*-hard. For brevity we do not include these results.

There are many ways to view approximations to these problems. A way different to the one explored in this paper, is to relax the restriction that one can generate the map exactly from the compression. Certainly, such languages would be more succinct than the ones defined in this paper; however it is not clear if shortest descriptions in these languages would be any easier to construct. What we know is that if one allows compressions from which we can generate maps that differ from the original map in at most k entries, the corresponding map compression problems are also *NP*-hard. This can be established by using arguments similar to those in this paper. Another way to view approximations is to find a decomposition in which the error (difference in absolute value between the true color and the generated color) is bounded by some constant. Of course, if the error is very small, the resulting problems are also *NP*-hard, but when the error is allowed to be large the approximation problem can be solved in polynomial time. It is interesting to investigate the trade off between quality of approximation (in a visual sense) and the time required to generate such solutions by different heuristic methods.

In this paper we concentrated on one and two dimensional maps. Another interesting problem, which is as hard as the ones discussed in this paper, is the three dimensional map compression problem. This problem has applications in terrain data as well as in "movies", which we defined as a sequence of two dimensional maps or frames. Compressions in this case would be important when there is not too much difference between adjacent

frames. Perhaps, for certain "movies" even simple heuristics could compress by a significant factor the amount of data needed to store or transmit this information.

References

[1] A. Apostolico and S. Hambrusch, New Clique and Independent Set Algorithms for Circle Graphs, *Disc. Appl. Math.,* **26**, March 1992, pp. 1 – 24. Correction/Addition in: *Disc. Appl. Math.,* **41**, January 1993, pp. 179 – 180.

[2] J. C. Culberson and R. A. Reckhow, Covering Polygons is Hard, *J. Algorithms,* **17**, July 1994, pp. 2 – 44.

[3] D. S. Franzblau, Performance Guarantees on a Sweep-Line Heuristic for Covering Rectilinear Polygons with Rectangles, *SIAM J. Disc. Math.,* **2**, August 1989, pp. 307 – 321.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* Freeman, San Francisco, 1979.

[5] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs,* Academic Press, New York, 1980.

[6] T. Gonzalez, Clustering to Minimize the Maximum Inter-Cluster Distance, *Theoretical Computer Science,* **38**, October 1985, pp. 293 – 306.

[7] G. Lu, Advances in Digital Image Compression Techniques *Computer Communications,* **16**, 1993, pp. 202 – 214.

[8] T. Ohtsuki, Minimum Dissection of Rectilinear Regions, in "Proceedings 1982 IEEE Symposium on Circuits and Systems," Rome, pp. 1210 - 1213.

[9] W. Lipski Jr and F. P. Preparata, Efficient Algorithms for Finding Maximum Matchings in Convex Bipartite Graphs and Related Problems, *Acta Informatica,* **15**, 1981, pp. 329 - 345.

[10] S.-Y. Wu and S. Sahni, Covering Rectilinear Polygons by Rectangles, *IEEE Transactions on CAD,* **9**, No. 2, April 1990.