Discrete Applied Mathematics 224 (2017) 80–90

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Computing the coarseness with strips or boxes

J.M. Díaz-Báñez [a], M.A. Lopez [b], C. Ochoa [c], P. Pérez-Lantero [d,*]

[a] Departamento Matemática Aplicada II, Universidad de Sevilla, Spain
[b] Department of Computer Science, University of Denver, USA
[c] Departamento de Ciencias de la Computación, Universidad de Chile, Chile
[d] Departamento de Matemática y Ciencia de la Computación, Universidad de Santiago, Chile

ARTICLE INFO

Article history:
Received 16 December 2015
Received in revised form 24 November 2016
Accepted 23 February 2017
Available online 22 March 2017

Keywords:
Coarseness
Rectangles
Strips
Computational geometry
Discrepancy

ABSTRACT

Recently, the concept of *coarseness* was introduced as a measure of how blended a 2-colored point set S is. In the definition of this measure, a convex partition Π, that is, a partition of S into sets $\{S_1, \ldots, S_k\}$ of S whose convex hulls are pairwise disjoint, is considered. The *discrepancy* of Π, denoted by $d(S, \Pi)$, is the smallest (bichromatic) discrepancy of the elements of Π. The *coarseness* of S, denoted by $\mathcal{C}(S)$, is then defined as the maximum of $d(S, \Pi)$ over all convex partitions Π of S. Roughly speaking, the value of the coarseness is high when we can split S into blocks, each with large discrepancy. It has been conjectured that computing the coarseness is NP-hard. In this paper, we study how to compute the coarseness for two constrained cases: (1) when the k elements of Π are separated by $k-1$ pairwise parallel lines (strips) and, (2) the case in which the cardinality of the partition is fixed and the elements of Π are covered by pairwise disjoint axis-aligned rectangles (boxes). For the first case we present an $O(n^2 \log^2 n)$-time algorithm, and show that such a computation problem is 3SUM-hard; for the second, we show that computing the coarseness with k boxes is NP-hard, when k is part of the input. For k fixed, we show that the coarseness can be computed in $O(n^{2k-1})$ time and propose more efficient algorithms for $k = 2, 3, 4$.

1. Introduction

For both theoretical and practical reasons, there is a large body of work considering the partition of point sets into clusters. A relevant question is whether a given 2-colored set may be separated into parts satisfying certain properties. The following question was posed in [5]: Given a bicolored point set S, is it possible to partition S so that every component of the partition can be considered as blue or red or, on the contrary, to decide that the colors are evenly distributed? In the second case, the points are called *well blended* and it would be pointless to apply a clustering algorithm.

The problem of assessing the homogeneity of a 2-colored set is related to the discrepancy theory [1,9] and arises naturally in different areas of computer science, such as computational learning theory, computational geometry, and computer graphics. For example, in supervised classification of machine learning we are given a labeled training set (for instance, a point set of samples, each point labeled as a positive or a negative sample) and we want to find a hypothesis (i.e. a partition) that can be used as a predictor for future query points [22,25]. In computational geometry, partitioning colored point sets

* Corresponding author.
E-mail addresses: dbanez@us.es (J.M. Díaz-Báñez), mlopez@cs.du.edu (M.A. Lopez), cochoa@dcc.uchile.cl (C. Ochoa), pablo.perez.l@usach.cl (P. Pérez-Lantero).

http://dx.doi.org/10.1016/j.dam.2017.02.022
0166-218X/

into monochromatic parts has already been considered by many authors (see e.g. [4,15,19,20]), and one application of these problems is the 2-class separability [21]. When we have two sets, consisting of, say, red points and blue points, a natural way to approach their separability is to look for a monochromatic convex partition in which the subsets forming the partition are as big as possible. For instance, when a 2-colored point set is linearly separable, we can partition it into two monochromatic parts. Finally, in computer graphics one promising approach is the application of the theory of discrepancy or irregularities of distribution [14,23].

When a clustering algorithm is considered on a 2-colored point set S, we are implicitly assuming that S can be partitioned into big monochromatic (or almost monochromatic) blocks. In other words, we expect to find blocks with high bichromatic discrepancy. Unfortunately, this is not possible when the two classes of points are blended. To detect well blended point sets, a reasonable parameter, the so called *coarseness* of S, has been recently introduced [5]. As pointed in [5], when one attempts to give a formal definition of well blended point sets, some contradictions and counterexamples may surface. Coarseness is based on the following idea: the data are not well blended if we are able to split the set into blocks, each with large discrepancy or, equivalently, the set is well blended if every partition into blocks contain an element with low discrepancy. The formal definition of coarseness considers the concept of *islands* [2] as blocks, as follows:

Definition 1. Let $S = R \cup B$ be a 2-colored set of n points in the plane. For a subset $Y \subseteq S$, define $\nabla(Y) = \big||R \cap Y| - |B \cap Y|\big|$. A nonempty subset I of S is called an *island* if there is a convex set C on the plane such that $I = C \cap S$. A *convex partition* of S is a partition of S into islands, with pairwise disjoint convex hulls. The *discrepancy* of a convex partition $\Pi = \{S_1, S_2, \dots, S_k\}$ of S, denoted $d(S, \Pi)$, is the minimum of $\nabla(S_i)$ for $i = 1, \dots, k$. The *coarseness* of S, denoted $\mathcal{C}(S)$, is defined as the maximum of $d(S, \Pi)$ over all the convex partitions Π of S.

In the case of point sets allowing big blocks of each color, an algorithm to compute the coarseness would naturally give us a clustering of S in which each cluster contains, as much as possible, a majority of one color. Otherwise, the optimal partition in the coarseness definition does not necessarily correspond to the best clustering. So, the coarseness problem is not a clustering algorithm but a measure to explore how the distribution of points is. Refer to [5] for a clear understanding of the concept of coarseness.

While the general convex partitions considered in the definition of coarseness can be viewed as generalizations of other geometric models, such as boxes, disks, etc., it is important to emphasize that the cardinality of the convex partition, k, is not fixed. This fact makes the coarseness computation a difficult problem. It is believed that the problem of computing the coarseness of a 2-colored point set is NP-hard [5], and an efficient approximation algorithm has recently been proposed [13]. On the other hand, when k is a fixed value, the coarseness of a bicolored point set can be computed in polynomial time by using a result of [7]. Using that any set of n points in the plane admits $O(n^{6k-12})$ k-partitions into disjoint islands, the coarseness of a 2-colored point set can be computed in $O(n^{6k-11})$ time, $k \geq 3$.

In this paper, we consider the problem of computing the coarseness of bicolored point sets for some special cases. We consider two types of islands in the convex partitions, namely, islands induced by strips or by rectangles with sides aligned with the coordinate axes. These types of geometric objects have been used in problems related to covering sets [4] and supervised clustering [16,6,24]. A remarkable difference is that in this paper we do not consider a covering of the classes but a convex partition following the definition of coarseness. We solve the open problem posed in [5] on the hardness of computing the coarseness for two cases: when the elements of the partition are induced by parallel strips, or by exactly k axis-aligned rectangles.

1.1. Definitions and notation

Let $\Pi = \{S_1, S_2, \dots, S_k\}$ be a convex partition of S. We say that Π is a *strip partition* if $k = 1$ or there are $k - 1$ parallel lines that simultaneously split S into the islands $S_1, S_2, \dots, S_k$. Since any line containing a point of S can be slightly translated so that the line does not contain any point of S, we assume $S \cap \ell = \emptyset$ for every line ℓ among the above $k-1$ parallel lines. For convenience, we further assume that such parallel lines divide the plane into k cells labeled so that S_i and S_{i+1} are contained in adjacent cells, respectively. In this sense, we say that S_i and S_{i+1} are *consecutive* islands of Π. The *strip coarseness* of S, denoted $\mathcal{C}_s(S)$, is the maximum of $d(S, \Pi)$ over all the strip partitions Π of S.

In this paper, all rectangles are assumed to be axis-aligned. Given any $X \subseteq S$, let $H(X)$ denote the minimum bounding rectangle (i.e. box hull) of X. We say that $\Pi = \{S_1, S_2, \dots, S_k\}$ is a *rectangle partition* of S if Π is a partition of S and the rectangles $H(S_1), H(S_2), \dots, H(S_k)$ are pairwise disjoint. The *rectangle coarseness* of S, denoted $\mathcal{C}_r(S)$, is the maximum of $d(S, \Pi)$ over all rectangle partitions Π of S and, given a fixed k, the *k-rectangle coarseness* of S is the maximum of $d(S, \Pi)$ over all the rectangle partitions Π of S into k islands.

Let $w : S \to \{-1, +1\}$ be the function such that $w(p) = +1$ if $p \in R$, and $w(p) = -1$ if $p \in B$. For every $X \subseteq S$, let $W(X) = \sum_{p \in X} w(p)$. Then, for every $X \subseteq S = R \cup B$ we have that $|X \cap R| - |X \cap B| = W(X)$ and $\nabla(X) = |W(X)|$. We say that (the color of) $X \subseteq S$ is *red* if X contains a strict majority of red points, i.e., if $W(X) > 0$, and that (the color of) X is *blue* if X contains a strict majority of blue points, i.e., if $W(X) < 0$. For a convex partition $\Pi = \{S_1, \dots, S_k\}$ of $S = R \cup B$, let $r_i = |R \cap S_i|$ and $b_i = |B \cap S_i|$, for each $i \in [1..k]$. For a given ordered list $p_1, p_2, \dots, p_n$ of the elements of S, and $s, t \in [1..n]$ with $s \leq t$, let $W(s, t) = W(\{p_s, \dots, p_t\}) = w(p_s) + \dots + w(p_t)$. Given a point $u \in \mathbb{R}^2$, let u_x and u_y denote the x- and y-coordinates of u, respectively.

1.2. Summary of results

In Section 2, we show that the problem of computing the strip coarseness of S is 3SUM-hard and present an algorithm that computes $\mathcal{C}_s(S)$ in $O(n^2 \log^2 n)$ time and $O(n)$ space. For the rectangle coarseness, $\mathcal{C}_r(S)$, we present several results. In one dimension (Section 3.1), if the input points are already sorted, $\mathcal{C}_r(S)$ can be computed in $O(n)$ time, and the k-rectangle coarseness can be computed in $O(n^2k)$ time for any $k \geq 2$, and in $O(n)$ time for $k = 2, 3, 4$. In Section 3.2, we consider the planar case and show that $\mathcal{C}_r(S)$ can be approximated by a constant factor. More precisely, we describe an $O(n \log n)$ time algorithm that computes a rectangle partition Π_{apx} such that

$$\max\Big\{\mathcal{C}_r(S)/8,\ \mathcal{C}_r(S)/4 - \nabla(S)\Big\} \leq d(S, \Pi_{apx}) \leq \mathcal{C}_r(S).$$

Furthermore, we show that computing the k-rectangle coarseness is NP-hard when k is part of the input and, for fixed $k \geq 1$, we show how to compute it in $O(n^{2k-1})$ time. For $k = 2, 3$, a faster $O(n \log n)$ time algorithm is given, and for $k = 4$, an $O(n^3)$-time algorithm.

2. Strip coarseness

The following results describe properties of the strip coarseness.

Lemma 2. *There exists an optimal strip partition Π of S such that either Π is the trivial partition $\{S\}$ or every pair of consecutive islands of Π have different colors.*

Proof. This lemma is a direct consequence of Lemma 7 in [5], which states that in a convex partition Π with a minimum number of islands and $d(S, \Pi) = \mathcal{C}(S)$, the existence of any two islands $S_i, S_j \in \Pi$ of equal colors implies that the number of islands of Π is at least three, and the convex hull of $S_i \cup S_j$ intersects the convex hull of some other island S_k of Π. Alternatively, the claim can be easily established by noticing that merging any pair of consecutive islands of equal colors cannot reduce the discrepancy of Π while, at the same time, reducing the number of islands. □

Using Lemma 2, we consider only optimal strip partitions in which every two consecutive islands have different colors.

Lemma 3. *There exists an optimal strip partition Π of S containing at most three islands.*

Proof. Let $\Pi = \{S_1, S_2, \ldots, S_k\}$ be an optimal strip partition of S with the minimum number of islands. For the sake of contradiction, suppose $k > 3$. Consider w.l.o.g. that $W(S_2) < 0$. If $\nabla(S_2) \leq \nabla(S_3)$ (i.e. $b_2 - r_2 \leq r_3 - b_3$) then

$$\begin{aligned}
\nabla(S_1) &= r_1 - b_1 \\
&\leq r_1 - b_1 + r_3 - b_3 - (b_2 - r_2) \\
&= r_1 + r_2 + r_3 - (b_1 + b_2 + b_3) \\
&= \nabla(S_1 \cup S_2 \cup S_3).
\end{aligned}$$

This implies that the strip partition $\Pi' = \{S_1 \cup S_2 \cup S_3, S_4, \ldots, S_k\}$ is also an optimal strip partition given that

$$\begin{aligned}
d(S, \Pi) &= \min\{\nabla(S_1), \ldots, \nabla(S_k)\} \\
&\leq \min\{\nabla(S_1), \nabla(S_4), \nabla(S_5), \ldots, \nabla(S_k)\} \\
&\leq \min\{\nabla(S_1 \cup S_2 \cup S_3), \nabla(S_4), \nabla(S_5), \ldots, \nabla(S_k)\} \\
&= d(S, \Pi'),
\end{aligned}$$

a contradiction, since Π was assumed to have the minimum number of islands. An analogous contradiction can be obtained if we assume that $\nabla(S_2) > \nabla(S_3)$. This implies that $\nabla(S_4) \leq \nabla(S_2 \cup S_3 \cup S_4)$. □

The following lemma characterizes the optimal strip partitions that have three elements.

Lemma 4. *Let $\Pi = \{S_1, S_2, S_3\}$ be an optimal strip partition of S. Then, $\nabla(S_2) \geq \nabla(S_1)$ and $\nabla(S_2) \geq \nabla(S_3)$, which is equivalent to $d(S, \Pi) = \min\{\nabla(S_1), \nabla(S_3)\}$.*

Proof. By Lemma 2, we can assume that S_1 and S_3 have equal colors. We may assume w.l.o.g. that S_1 and S_3 are red, while S_2 is blue. For the sake of contradiction, suppose that $\nabla(S_2) < \nabla(S_1)$, that is, $b_2 - r_2 < r_1 - b_1$ which is equivalent to $r_1 + r_2 - (b_1 + b_2) > 0$. Then, we have

$$\begin{aligned}
d(S, \{S\}) &= \nabla(S) \\
&= r_1 + r_2 + r_3 - (b_1 + b_2 + b_3) \\
&= (r_3 - b_3) + r_1 + r_2 - (b_1 + b_2) \\
&> r_3 - b_3 \\
&= \nabla(S_3) \\
&\geq \min\{\nabla(S_1), \nabla(S_2), \nabla(S_3)\} \\
&= d(S, \Pi).
\end{aligned}$$

The fact $d(S, \{S\}) > d(S, \Pi)$ contradicts the optimality of Π. A similar contradiction can be obtained by assuming that $\nabla(S_2) < \nabla(S_3)$. The result thus follows. $\square$

Lemma 5. *Suppose that the elements of S are located on a horizontal line and denoted $p_1, p_2, \ldots, p_n$ from left to right. If the strip partition $\Pi = \{\{p_1, \ldots, p_i\}, \{p_{i+1}, \ldots, p_{j-1}\}, \{p_j, \ldots, p_n\}\}$ ($i \in [1..n-2]$ and $j \in [i+2..n]$) is optimal and $\{p_1, \ldots, p_i\}$ is red, then i and j maximize*

$$\min\left\{w(p_1) + \cdots + w(p_i), w(p_j) + \cdots + w(p_n)\right\}.$$

Proof. Assume w.l.o.g. that

$$W(1, i) = \min\{W(1, i), W(j, n)\}.$$

Note that $W(1, i) = \nabla(\{p_1, \ldots, p_i\}) = d(\Pi, S)$ by Lemma 4. For the sake of contradiction, suppose that there exist indices $i' \in [1..n-2]$ and $j' \in [i'+2..n]$ such that

$$\min\left\{W(1, i'), W(j', n)\right\} > W(1, i).$$

Since Π is optimal, the strip partition $\Pi' = \{\{p_1, \ldots, p_{i'}\}, \{p_{i'+1}, \ldots, p_{j'-1}\}, \{p_{j'}, \ldots, p_n\}\}$, where $\{p_1, \ldots, p_{i'}\}$ and $\{p_{j'}, \ldots, p_n\}$ are red, must satisfy $d(S, \Pi') \le d(S, \Pi)$, which implies

$$\nabla(\{p_{i'+1}, \ldots, p_{j'-1}\}) \le \max\left\{\nabla(\{p_1, \ldots, p_{i'}\}), \nabla(\{p_{j'}, \ldots, p_n\})\right\}$$

which can be rewritten as

$$\left|W(i'+1, j'-1)\right| \le \max\left\{W(1, i'), W(j', n)\right\}$$

since $\nabla(\{p_1, \ldots, p_{i'}\}) = W(1, i')$, $\nabla(\{p_{i'+1}, \ldots, p_{j'-1}\}) = |W(i'+1, j'-1)|$, and $\nabla(\{p_{j'}, \ldots, p_n\}) = W(j', n)$. This, in turn, implies

$$\begin{aligned}
d(S, \{S\}) &= \left|W(1, i') + W(i'+1, j'-1) + W(j', n)\right| \\
&\ge \left|W(1, i')\right| - \left|W(i'+1, j'-1)\right| + \left|W(j', n)\right| \\
&= W(1, i') + W(j', n) - \left|W(i'+1, j'-1)\right| \\
&= \min\left\{W(1, i'), W(j', n)\right\} + \max\left\{W(1, i'), W(j', n)\right\} - \left|W(i'+1, j'-1)\right| \\
&\ge \min\left\{W(1, i'), W(j', n)\right\} \\
&> W(1, i) \\
&= d(\Pi, S),
\end{aligned}$$

contradicting the optimality of Π. The lemma thus follows. $\square$

By using the above lemmas, we obtain the main result of this section: an $O(n^2 \log^2 n)$-time algorithm to compute the strip coarseness of a 2-colored point set S. The algorithm is described in the proof of the following theorem.

Theorem 6. *Given a 2-colored point set S of n points in the plane, the strip coarseness of S can be computed in $O(n^2 \log^2 n)$ time and $O(n)$ space.*

Proof. By Lemma 3, we compute the three values d_1, d_2, and d_3 where: $d_1 = d(S, \{S\})$; d_2 is the maximum of $d(S, \{S_1, S_2\})$ over all the strip partitions $\{S_1, S_2\}$ of S; and d_3 is the maximum of $d(S, \{S_1, S_2, S_3\})$ over all the strip partitions $\{S_1, S_2, S_3\}$ of S. Finally, we return $\max\{d_1, d_2, d_3\}$ as the value of $\mathcal{C}_s(S)$. Computing d_1 can trivially be done in $O(n)$ time. The value d_2 is what Bereg et al. [5] called the *linear coarseness* and showed how to compute it in $O(n^2)$ time. Furthermore, they proved that computing d_2 is 3SUM-hard. The rest of the proof is devoted to showing that d_3 can be computed in $O(n^2 \log^2 n)$ time.

By Lemma 2, we can first find an optimal strip partition $\Pi = \{S_1, S_2, S_3\}$ subject to S_1 and S_3 are red and S_2 is blue. Similarly, we can find an optimal strip partition subject to S_1 and S_3 are blue and S_2 is red. Since the two cases are symmetric, it suffices to focus on the first one.

Every strip partition $\Pi = \{S_1, S_2, S_3\}$ is induced by two parallel lines ℓ_1 and ℓ_3, directed in the same direction, where S_1 is to the left of ℓ_1, S_2 is in between ℓ_1 and ℓ_3, and S_3 is to the right of ℓ_3. Let ℓ be any directed line such that no line through two points of S is perpendicular to ℓ, and label the input points $p_1, p_2, \ldots, p_n$, sorted in the direction of the line ℓ (i.e., according to their orthogonal projections onto ℓ). We show now how to compute the discrepancy of the optimal strip partition Π_ℓ induced by ℓ_1 and ℓ_3 when ℓ_1 and ℓ_3 are restricted to be perpendicular to ℓ.

The strip partition Π_ℓ has the form $\{\{p_1, \ldots, p_i\}, \{p_{i+1}, \ldots, p_{j-1}\}, \{p_j, \ldots, p_n\}\}$, where the indices $i \in [1..n-2]$ and $j \in [i+2, n]$ maximize $\min\{W(1, i), W(j, n)\}$ (Lemma 5). Observe that there exists an index $t \in [2..n-1]$ such that $i < t < j$. Furthermore, for such an index t we have that $W(1, i) = mp(1, t-1)$ and $W(j, n) = ms(t+1, n)$, where for every pair of indices $i, j \in [1..n]$, $i \le j$, $mp(i, j) = \max_{s \in [i..j]} W(i, s)$ and $ms(i, j) = \max_{t \in [i..j]} W(t, j)$.

Such an index t can be found by using a binary search in the range $[2..n-1]$. Namely, given a value of t, we compute both $mp(1, t-1)$ and $ms(t+1, n)$. Then, since the sequence $\{\max_{s\in[1..i]} W(1, s)\}_{i=1}^{n}$ is ascending and the sequence $\{\max_{s\in[i..n]} W(s, n)\}_{i=1}^{n}$ is descending, we perform the following procedure for the current value of t: If $mp(1, t-1) = ms(t+1, n)$, then we have that $d(S, \Pi_\ell) = \min\{mp(1, t-1), W(1, n) - mp(1, t-1) - ms(t+1, n), ms(t+1, n)\}$ and stop the search. Otherwise, we increase t if $mp(1, t-1) < ms(t+1, n)$ and decrease t if $mp(1, t-1) > ms(t+1, n)$, and continue with the search. If there is no range for t to continue the search, then $d(S, \Pi_\ell) = \min\{mp(1, t-1), ms(t+1, n), W(1, n) - mp(1, t-1) - ms(t+1, n)\}$ for any value of t considered in the search that maximizes $\min\{mp(1, t-1), ms(t+1, n)\}$.

We can use the MCS-tree of Cortés et al. [11] to store the sequence $w(p_1), w(p_2), \ldots, w(p_n)$. The MCS-tree is a tree-like linear-size data structure that maintains a sequence of n numbers over updates of its elements so that the update of any element costs $O(\log n)$ time, and range maximum sum queries can be answered in $O(\log n)$ time. In particular, given an index $t \in [2..n-1]$, a MCS-tree over the sequence $w(p_1), w(p_2), \ldots, w(p_n)$ allows us to compute the sums $mp(1, t-1)$ and $ms(t+1, n)$ in $O(\log n)$ time. Then, the binary search can run in $O(\log^2 n)$ time.

The overall algorithm to compute d_3 is then as follows: We fix a line ℓ, build the MCS-tree over the sequence $w(p_1), w(p_2), \ldots, w(p_n)$ induced by ℓ, and compute $d(S, \Pi_\ell)$ with the binary search. Then, we perform a rotational sweep with the line ℓ in which we maintain the value of $d(S, \Pi_\ell)$ for each change of the sequence $w(p_1), w(p_2), \ldots, w(p_n)$. There are $O(n^2)$ changes in total and each of them can be enacted by performing a swap of two adjacent elements whose corresponding points in S define a line perpendicular to ℓ. Each swap can be implemented with two $O(\log n)$-time one-element updates in the MCS-tree. After the MCS-tree is updated, a new value of $d(S, \Pi_\ell)$ is computed with the binary search in $O(\log^2 n)$ time. The value $d_3 = \max_\ell d(S, \Pi_\ell)$ over all the positions of ℓ is returned.

To perform the rotational sweep and process each swap in the sequence $w(p_1), w(p_2), \ldots, w(p_n)$, we iterate over the set L of at most $\binom{n}{2} = O(n^2)$ lines through two points of S sorted by slope. To do so, we dualize the points in S to the set of lines S' so that the lines of L map to the intersection points between the lines of S', and the left-to-right order of such points maps precisely to the lines of L sorted by slope [12]. The intersection points between the lines of S' can be recognized from left to right with the plane sweep algorithm for segment intersection [12], which runs in $O(n \log n + s \log n) \subseteq O(n^2 \log n)$ time, where $s = O(n^2)$ is the number of intersections, and $O(n)$ space. For each intersection point recognized, we perform the corresponding swap in the sequence $w(p_1), w(p_2), \ldots, w(p_n)$ and apply the binary search. Since we apply the binary search for each of the $O(n^2)$ swaps, the overall algorithm runs in $O(n^2 \log^2 n)$ time. □

Corollary 7. *Given a set S of colored points on a line, the strip coarseness of S can be computed in $O(n)$ time if the sorted list of points along the line is given, or in $O(n \log n)$ time otherwise.*

We finish this section by proving that computing the strip coarseness is 3SUM-hard. The proof uses the construction of Bereg et al. [5] that shows that computing the linear coarseness, that is, the maximum discrepancy of partitions with two subsets of the point set, is 3SUM-hard.

Theorem 8. *Given a 2-colored point set S of n points, computing the strip coarseness of S is 3SUM-hard.*

Proof. Let d_2 be the maximum of $d(S, \{S_1, S_2\})$ over all the strip partitions $\{S_1, S_2\}$, and d_3 the maximum of $d(S, \{S_1, S_2, S_3\})$ over all the strip partitions $\{S_1, S_2, S_3\}$ of S. Bereg et al. [5] proved that, for every integer $d \geq 1$, deciding $d_2 = d$ is 3SUM-hard, making a reduction from an instance $\{x_1, x_2, \ldots, x_n\}$ of 3SUM to a 2-colored point set S. We will show in what follows that for $d = 2$ such a point set S satisfies $d_2 \geq d_3 \geq 0 = d(S, \{S\})$. Thus, computing $\mathcal{C}_s(S) = d_2$ is also 3SUM-hard. Given an instance $\{x_1, x_2, \ldots, x_n\}$ of 3SUM, Bereg et al. [5] build the point set $S = R \cup B$, where $R = \{p_i = (x_i - \varepsilon, x_i^3) : i = 1, \ldots, n\}$ and $B = \{q_i = (x_i + \varepsilon, x_i^3) : i = 1, \ldots, n\}$ for some small enough $\varepsilon > 0$. Consider the n red–blue pairs (p_i, q_i), $i \in [1..n]$. We say that a line *separates* a pair if the red element is to one of side of the line and the blue one is to the other side. Note that if a line separates more than one pair, then the red (resp. blue) elements of the pairs are on the same side of the line. Furthermore, ε satisfies: a line separates three pairs (p_i, q_i), (p_j, q_j), and (p_k, q_k) for different $i, j, k \in [1..n]$ if and only if $x_i + x_j + x_k = 0$; for every two different pairs there exists a line that separates them; and there does not exist any line separating at least four different pairs. Under these conditions, $d_2 \in \{2, 3\}$, and $d_2 = 3$ if and only if three different pairs are separated by the same line. Let $\Pi = \{S_1, S_2, S_3\}$ be a strip partition of S, induced by the parallel lines ℓ_1 and ℓ_2, such that $d_3 = d(S, \Pi)$. Since the pairs separated by ℓ_1 or ℓ_2 are the only ones that count to compute $\nabla(S_1)$, $\nabla(S_2)$, and $\nabla(S_3)$, then we have that $\nabla(S_1), \nabla(S_3) \leq 3$ which implies $d_3 \leq 3$. Furthermore, $\nabla(S_1) = \nabla(S_3) = 3$ implies $\nabla(S_2) = 0$. Then, $2 = d_2 < d_3 = 3$ cannot happen. Consequently, we always have $d_3 \leq d_2$, and the result follows. □

3. Rectangle coarseness

3.1. One dimension

Let $p_1, p_2, \ldots, p_n$ denote the elements of S, sorted from left to right, and consider the sequence $w(p_1), w(p_2), \ldots, w(p_n)$. Observe that, in 1D, the rectangle coarseness and the strip coarseness are equivalent. Consequently, by Corollary 7, this value can be computed in $O(n)$ time, if the input points are given in sorted order, or in $O(n \log n)$ time otherwise. In the following, we describe algorithms to compute the k-rectangle coarseness, for fixed k.

Theorem 9. *The k-rectangle coarseness in one dimension can be computed in $O(n^2k)$ time and $O(nk)$ space.*

Proof. We solve this problem by means of dynamic programming. Namely, let $OPT(j,t)$ denote the t-rectangle coarseness of the point set $\{p_1, p_2, \ldots, p_j\}$ for every $j, t \in [1..n]$. $OPT(j,t)$ satisfies the following recurrence:

$$OPT(j,t) = \begin{cases} 0 & j < t \\ |W(1,j)| & t = 1 \\ \max\limits_{s\in[1..j-1]} \min\{OPT(s,t-1), |W(s+1,j)|\} & 1 < t \le j. \end{cases}$$

Preprocessing $w(p_1), w(p_2), \ldots, w(p_n)$ by computing all the prefix sums $W(1,1), W(1,2), \ldots, W(1,n)$, each sum $W(s+1,j) = W(1,j) - W(1,s)$ can be computed in constant time. This implies that the k-rectangle coarseness of S is equal to $OPT(n,k)$ and can be computed in $O(n^2k)$ time and $O(nk)$ space. $\square$

We consider now the k-rectangle coarseness of S in 1D for the special cases $k = 2, 3, 4$. In each case, we give an $O(n)$-time algorithm by assuming that the sequence $w(p_1), w(p_2), \ldots, w(p_n)$ is given. If this sequence is not given, it can be obtained in $O(n \log n)$ time by sorting the elements of S, and the k-rectangle coarseness then computed in $O(n \log n)$ time.

Proposition 10. *The 2-rectangle coarseness of the set $\{p_i, p_{i+1}, \ldots, p_j\}$ is given by a convex partition of the form $\{\{p_i, \ldots, p_s\}, \{p_{s+1}, \ldots, p_j\}\}$ that satisfies the following properties:*

1 *If $W(i,s) > 0$ and $W(s+1,j) \le 0$, then $W(i,s) = mp(i,j)$.*
2 *If $W(i,s) \le 0$ and $W(s+1,j) > 0$, then $W(s+1,j) = ms(i,j)$.*
3 *If $W(i,s), W(s+1,j) > 0$, $W(i,s), W(s+1,j) < 0$, or $\{W(i,s), W(s+1,j)\} = \{-1, 0\}$ then*

$$\Big\{W(i,s), W(s+1,j)\Big\} = \left\{\left\lfloor \frac{|W(i,j)|}{2} \right\rfloor, \left\lceil \frac{|W(i,j)|}{2} \right\rceil\right\}.$$

Proof. The proof can be derived from Lemma 12 in [5]. $\square$

Theorem 11. *The k-rectangle coarseness in one dimension and $k = 2, 3, 4$ can be computed in $O(n)$ time if we know the left-to-right order of S. Otherwise, it can be computed in $O(n \log n)$ time.*

Proof. Using the data structure of Chen and Chao [10], we can preprocess $w(p_1), w(p_2), \ldots, w(p_n)$ in linear time so that, for any four indices $i \le j \le i' \le j'$, the following query can be answered in constant time: compute the maximum sum $W(s,s')$ such that $i \le s \le j$ and $i' \le s' \le j'$. Notice that, if $i = j = i'$, then $W(s,s') = W(i,s') = mp(i,j')$, and that if $j = i' = j'$, then $W(s,s') = W(s,j) = ms(i,j)$.

Using the above preprocessing and Proposition 10, the 2-rectangle coarseness $c_2(i,j)$ of every set of the form $\{p_i, p_{i+1}, \ldots, p_j\}$ can be computed in $O(1)$ time. This gives us $c_2(1,n)$, the 2-rectangle coarseness of S, in the same time bound. To compute the 3-rectangle coarseness, we return

$$\max_{i\in[2..n-1]} \min\Big\{c_2(1,i), |W(i+1,n)|\Big\}.$$

Similarly, to compute the 4-rectangle coarseness, we return

$$\max_{i\in[2..n-2]} \min\Big\{c_2(1,i), c_2(i+1,n)\Big\}.$$

Each of these two maximum values can be computed in $O(n)$ time. The result follows. $\square$

3.2. Two dimensions

Computing the rectangle coarseness of a 2-colored point set S seems to be as hard as computing the coarseness [5,13]. Similar to the approximate computation of the coarseness given by Díaz-Báñez et al. [13], we show an approximate computation of the rectangle coarseness with ratio between 1/8 and 1/4, using a notion that we call 2-separable island. An island $S' \subseteq S$ is 2-*separable* if there exists a quadrant (i.e. the intersection of two axis-parallel halfplanes) that contains S' and its complement (the union of three quadrants) contains $S \setminus S'$. The proof of the following lemma describes an algorithm to find a 2-separable island $S' \subseteq S$ maximizing $\nabla(S')$. This algorithm can be derived from the techniques of Cortés et al. [11], combining plane sweeps and the MCS-tree, in order to find a rectangle H that maximizes $W(H)$. We describe it here for completeness and facilitate understanding of some of the algorithms in this section which use similar techniques.

Lemma 12. *Let $S = R \cup B$ be 2-colored point set in the plane. A 2-separable island S' maximizing $\nabla(S')$ can be found in $O(n \log n)$ time.*

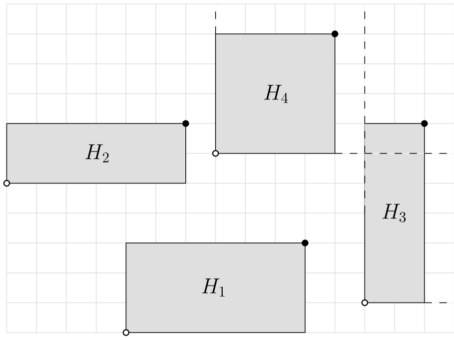

Fig. 1. The four rectangles H_1, H_2, H_3, and H_4 of the rectangle partition $\{S_1, S_2, S_3, S_4\}$: $H_1 \prec H_3$ and $H_2 \prec H_4$. Since H_3 and H_4 are the maximal elements in $\prec$, then $(M_1, M_2) = (H_4, H_3)$. Note that $M_2 = H_3$ can be separated from the other rectangles with an upper-right quadrant.

Proof. We show how to find a 2-separable island S' that maximizes $W(S')$ such that there exists an upper-right quadrant Q satisfying $S' = S \cap Q$. The other cases where Q is upper-left, lower-left, or lower-right, and/or $-W(S')$ is maximized, can be solved symmetrically. Let $p_1, p_2, \ldots, p_n$ denote the elements of S sorted in descending order of y-coordinate. Let ℓ be a vertical line through or to the left of at least one point of S, and let $w_\ell : S \to \{-1, 0, +1\}$ be the following weight function dependent on ℓ: If $p \in S$ is to the left of the line ℓ then $w_\ell(p) = 0$. Otherwise, if p lies on or to the right of ℓ, then $w_\ell(p) = w(p)$. This choice of function allows us to "discard" the points to the left of ℓ by assigning them weight zero. Consider now the sequence $w_\ell(p_1), w_\ell(p_2), \ldots, w_\ell(p_n)$, and observe that an upper-right quadrant Q_ℓ bounded to the left by ℓ that maximizes $W(Q_\ell \cap S)$ is bounded below by the horizontal line through a point p_i, $i \in [1..n]$ that maximizes $W_\ell(1, i) = w_\ell(p_1) + \cdots + w_\ell(p_i)$. Furthermore, $W(Q_\ell \cap S) = W_\ell(1, i)$.

The algorithm is then as follows: We start with the line ℓ to the left of all points of S, and sweep S from left to right with ℓ, keeping track of the maximum of $W(Q_\ell \cap S)$. The sequence $w_\ell(p_1), w_\ell(p_2), \ldots, w_\ell(p_n)$ is stored in an MCS-tree [11], such that every time the line ℓ crosses a point p_j of S, $j \in [1..n]$, the element $w_\ell(p_j)$ is updated in $O(\log n)$ time in the MCS-tree. After the update is performed, an index $i \in [1..n]$ that maximizes $W_\ell(1, i)$ can be found in $O(1)$ time at the root of the MCS-tree. The 2-separable island S' returned is the set $\{p_j : j \in [1..i], w_\ell(p_j) \neq 0\}$ where $W(Q_\ell \cap S) = W_\ell(1, i)$ is maximum. The running time is clearly $O(n \log n)$. $\square$

The following lemma shows that every rectangle partition contains a 2-separable island. Given two points $u, v \in \mathbb{R}^2$, we write $u \prec v$ (or say that v *dominates* u) if $u_x \le v_x$ and $u_y \le v_y$.

Lemma 13. *Let $S = R \cup B$ be 2-colored point set in the plane. Every rectangle partition of S contains a 2-separable island, which can be separated with a quadrant in any given direction.*

Proof. Let $\{S_1, S_2, \ldots, S_k\}$ be a rectangle partition of S, and let $H_i = H(S_i)$ for every $i \in [1..k]$. We will prove that there exists an upper-right quadrant that contains a rectangle of the set $\{H_1, H_2, \ldots, H_n\}$ and does not intersect any other rectangle of this set. This implies the lemma. For the other quadrant directions the proof is similar by symmetry.

Given a rectangle H, let $v(H)$ denote the bottom-left vertex of H, and $Q(H)$ the upper-right quadrant with origin $v(H)$ (which is the inclusion minimum such a quadrant that contains H). For every $i \neq j$, we write $H_i \prec H_j$ if and only if $v(H_i)_x \le v(H_j)_x$ and $v(H_i)_y \le v(H_j)_y$. That is, vertex $v(H_j)$ dominates (in the vectorial sense) vertex $v(H_i)$. Observe that $\prec$ is a partial order on the rectangles $H_1, H_2, \ldots, H_k$, thus there exists at least one maximal rectangle with respect to $\prec$. Let $\{M_1, M_2, \ldots, M_t\} \subseteq \{H_1, H_2, \ldots, H_n\}$ be the set of maximal rectangles with respect to $\prec$, and assume $v(M_1)_x < v(M_2)_x < \cdots < v(M_t)_x$ through labeling (refer to Fig. 1). We prove by induction on t that there exists an upper-right quadrant that contains a rectangle of the set $\{M_1, M_2, \ldots, M_t\}$ and does not intersect any other rectangle. If $t = 1$ this statement is trivially true. Then, assume $t \ge 2$. If $Q(M_1)$ does not intersect any rectangle in $\{M_2, \ldots, M_t\}$, then $Q(M_1)$ can be the desired quadrant. Otherwise, let $i \in [2..t]$ be the minimum index such that $Q(M_1) \cap M_i$ is not empty. Observe that the vertical line through $v(M_i)$ separates the rectangles $M_1, M_2, \ldots, M_{i-1}$ from the rectangles $M_i, M_{i+1}, \ldots, M_t$. By the inductive hypothesis, there is an upper-right quadrant containing a rectangle in $\{M_i, M_{i+1}, \ldots, M_t\}$ and does not intersect any other rectangle of this set. By the former argument, such a quadrant neither intersects any rectangle among $M_1, M_2, \ldots, M_{i-1}$. This completes the proof. $\square$

Theorem 14. *Let $S = R \cup B$ be 2-colored point set in the plane. In $O(n \log n)$ time a rectangle partition Π_{apx} of S can be computed such that*

$$\max\left\{\mathcal{C}_r(S)/8,\ \mathcal{C}_r(S)/4 - \nabla(S)\right\} \le d(S, \Pi_{apx}) \le \mathcal{C}_r(S).$$

Proof. Let $S' \subseteq S$ be a 2-separable island that maximizes $\nabla(S') = |W(S')|$, which can be computed in $O(n \log n)$ time by Lemma 12. Díaz-Báñez et al. [13] proved that if there exists an island I of S such that $\nabla(I) \ge t$ for some t, and I is contained in the intersection of two halfplanes whereas $S \setminus I$ is contained in the complement of that intersection, then there exists a convex partition Π such that $d(S, \Pi) \geq \max\{t/8, t/4 - \nabla(S)\}$. The convex partition Π is obtained by splitting S into two or four islands by using one or two of the boundary lines of such halfplanes, respectively. Since S' is 2-separable, we can take the two axis-parallel halfplanes defining the quadrant that contains S' and completely excludes $S \setminus S'$, and use them in the arguments of Díaz-Báñez et al. [13] to show that it is possible to give a rectangle partition Π_{apx} such that

$$\max\left\{\nabla(S')/8,\ \nabla(S')/4 - \nabla(S)\right\} \leq d(S, \Pi_{apx}).$$

Since every rectangle partition Π has a 2-separable island (Lemma 13), we have that $d(S, \Pi) \leq \nabla(S')$ for every Π, which implies

$$\max\left\{\mathcal{C}_r(S)/8,\ \mathcal{C}_r(S)/4 - \nabla(S)\right\} \leq d(S, \Pi_{apx}).$$

The result follows. □

We now turn our attention to k-rectangle coarseness. There are two cases. When k is part of the input, we show that its computation is NP-hard. When k is fixed, we show how to compute it in $O(n^{2k-1})$ time. For $k = 2, 3, 4$, more efficient algorithms are given.

Theorem 15. *Given a 2-colored point set $S = R \cup B$ in the plane with n points, and a value $k \in [1..n]$, it is NP-hard to compute the k-coarseness of S.*

Proof. We will reduce from the Perfect Monochromatic Rectangle Matching (PMRM) problem which is NP-complete [8]. Given a 2-colored point set in the plane, the PMRM problem asks whether there exists a set of pairwise disjoint rectangles such that each rectangle covers precisely two points of the same color, and that each point is covered by some rectangle. The PMRM problem is NP-complete even if the points are in general position. Given an instance $S = R \cup B$ of the PMRM problem, the answer is affirmative if and only if for $k = \lceil |R|/2 \rceil + \lceil |B|/2 \rceil$ the k-rectangle coarseness of S is equal to two, that is, $\mathcal{C}_r(R \cup B) = 2$. □

Theorem 16. *Given a 2-colored point set $S = R \cup B$ in the plane with n points, and a fixed value $k \in [1..n]$, the k-rectangle coarseness of S can be computed in $O(n^{2k-1})$ time.*

Proof. By Lemma 13, we can assume through labeling, that any rectangle partition $\{S_1, S_2, \dots, S_k\}$ of S satisfies the following property: For $i = 1, \dots, k$, the set S_i is a 2-separable island of the set $S_i \cup S_{i+1} \cup \cdots \cup S_k$, which can be separated using an upper-right quadrant. Accordingly, in the rest of the proof, we assume, without restating this property, that every 2-separable island can be separated using an upper-right quadrant. For every $X \subseteq S$, let $Q(X)$ denote the minimum upper-right quadrant that covers X. The algorithm enumerates recursively all rectangle partitions $\{S_1, S_2, \dots, S_k\}$, that is, for each selection of $S_1, S_2, \dots, S_{j-1}$, $j \in [1..k-1]$, it iterates over all possible islands S_j (together with $\nabla(S_j)$) such that the following conditions are satisfied:

(1) S_j is a 2-separable island of the set $S \setminus (S_1 \cup \cdots \cup S_{j-1})$.
(2) For each $t \in [1..j-1]$, we have $H(S_j) \cap Q(S_t) = \emptyset$.

Both conditions are used to guarantee that, with $j - k$, all k-rectangle partitions $\{S_1, S_2, \dots, S_k\}$ are enumerated. Each island S_i is represented in $O(1)$ space by $H(S_i)$. We now show that the above enumeration can be done in $O(n^{2k-1})$ time. Assume that the elements of S have been sorted twice, first by x-coordinate and second by y-coordinate.

For $j = 1$, all the $O(n^2)$ pairs $(S_1, \nabla(S_1))$, where $S_1 \subseteq S$ is a 2-separable island of S, can be enumerated in overall $O(n^2)$ time, by using the x- and y-orderings of S.

For $j \in [2..k-1]$, let S_j be a 2-separable island of $S \setminus (S_1 \cup \cdots \cup S_{j-1})$. Note that condition (2) is satisfied if and only if the upper-right vertex of $H(S_j)$ is not in the region $Q(S_1) \cup \cdots \cup Q(S_{j-1})$, whose boundary is precisely the upper-right staircase defined by the minima of the lower-left vertices of the rectangles $H(S_1), \dots, H(S_{j-1})$. Using the x- and y-order of S, the vertices of such a staircase can be sorted by x- or y-coordinate in $O(n)$ time. Then, using the x- and y-orders of both S and the vertices of the staircase, all valid pairs $(S_j, \nabla(S_j))$, where S_j satisfies conditions (1) and (2), can be iterated in $O(n^2)$ time.

For $j = k$, the only island to iterate is $S_k = S \setminus (S_1 \cup \cdots \cup S_{k-1})$, which can be done in $O(n)$ time with arguments similar to that of the above two cases.

The overall running time is then $O(n^{2k-1})$. □

Theorem 17. *Let $S = R \cup B$ be a 2-colored point set in the plane with n points. For $k = 2, 3$, the k-rectangle coarseness of S can be computed in $O(n \log n)$ time.*

Proof. For $k = 2$, the islands S_1 and S_2 can always be separated by an axis-parallel line. Then, computing the 2-rectangle coarseness of S reduces to computing the 2-rectangle coarseness of two point sets in one dimension, and returning the maximum between them, which can be done in $O(n \log n)$ time (Theorem 11). Such point sets are: the points of S sorted by x-coordinate, and the points of S sorted by y-coordinate.

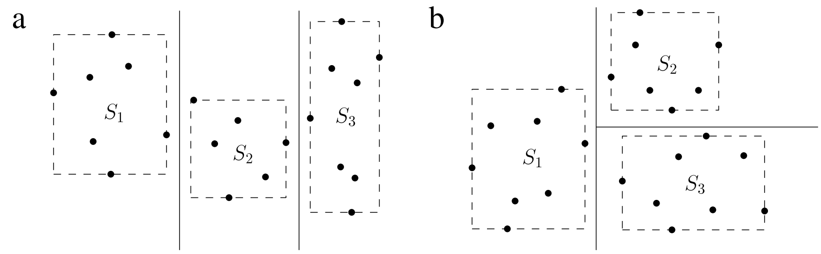

Fig. 2. The islands S_1, S_2, and S_3 are separated by either: (a) two axis-parallel non-intersecting lines, or (b) an axis-parallel line and an axis-parallel half-line.

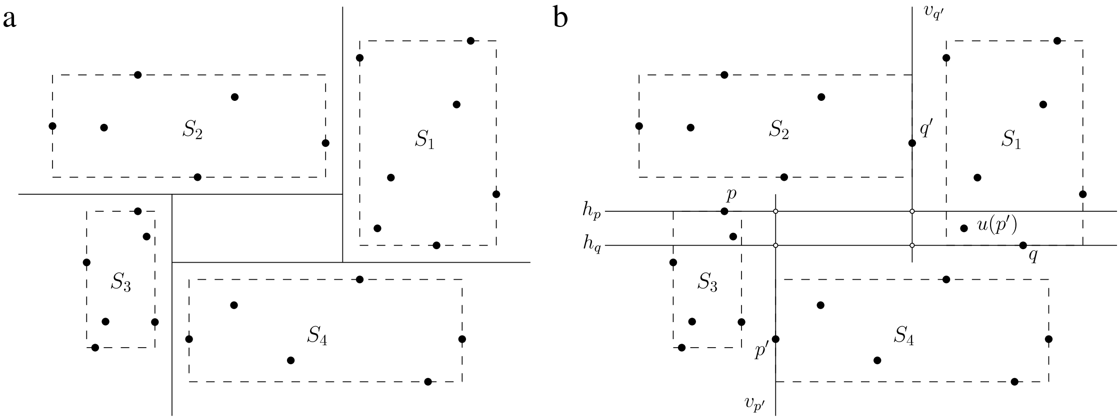

Fig. 3. The islands S_1, S_2, S_3, and S_4 are separated by four axis-parallel halflines.

For $k = 3$, S_1, S_2, and S_3 can always be separated by two axis-parallel non-intersecting lines (Case 1, Fig. 2(a)), or an axis-parallel line and an axis-parallel halfline orthogonal to the line with the apex on it (Case 2, Fig. 2(b)). For Case 1, computing the 3-rectangle coarseness reduces to computing such a coarseness in two one-dimensional point sets, the x-order of S and the y-order of S. Then, the 3-rectangle coarseness can be computed in $O(n \log n)$ time by Theorem 11.

For Case 2, assume w.l.o.g. the relative positions of S_1, S_2, and S_3 shown in Fig. 2(b). The other configurations for S_1, S_2, and S_3 can be treated analogously. We sweep S from left to right with a vertical line ℓ, and at every event in which ℓ crosses a point of S, the set $S_1 \subseteq S$ contains the points to the left ℓ, and $\{S_2, S_3\}$ maximizes $d(S \setminus S, \{S_2, S_3\})$, that is, $d(S \setminus S, \{S_2, S_3\})$ is equal to the 2-rectangle coarseness of the one-dimensional y-order of the points $S \setminus S_1$ to the right of ℓ_1. We maintain the maximum of $\nabla(S_1)$, $\nabla(S_2)$, and $\nabla(S_3)$ during the sweep. At each event, $\nabla(S_1)$ can be updated in $O(1)$ time, and $d(S \setminus S, \{S_2, S_3\})$ in $O(\log n)$ time by using an MCS-tree over the y-order of S with the weight function $w_\ell(p)$ (that given $p \in S$ returns $w(p)$ if p is to the right of ℓ, and 0 otherwise, see Lemma 13), and Proposition 10. The running time is $O(n \log n)$. □

Theorem 18. *Let $S = R \cup B$ be a 2-colored point set S in the plane with n points. The 4-rectangle coarseness of S can be computed in $O(n^3)$ time.*

Proof. Observe that the minimum of $d(S, \Pi)$ over all rectangle partitions $\Pi = \{S_1, S_2, S_3, S_4\}$, such that there exists an axis-parallel line separating two islands from the other two ones, can be computed in $O(n \log n)$ time with an algorithm similar to that given in Theorem 17 for computing the 3-rectangle coarseness. Namely, we can sweep S with an axis-parallel line and dynamically maintain the 2-rectangle coarseness of the points to one side of the line, and the 2-rectangle coarseness of the points to the other side.

Also note that the minimum of $d(S, \Pi)$ over all rectangle partitions $\Pi = \{S_1, S_2, S_3, S_4\}$, such that there exists an axis parallel line separating one island from the other three ones, can be computed in $O(n^2 \log n)$ time by computing the 3-rectangle coarseness of $O(n)$ subsets of S.

In the following, we show how to compute in $O(n^3)$ time the maximum of $d(S, \Pi)$ over all rectangles partitions $\Pi = \{S_1, S_2, S_3, S_4\}$ for which there is no axis-parallel line separating one or two islands from the other ones. Assume w.l.o.g. that the relative positions of $H(S_1)$, $H(S_2)$, $H(S_3)$, and $H(S_4)$ are as shown in Fig. 3(a), being separated by four axis-parallel halflines.

Given a point $a \in \mathbb{R}^2$, let h_a and v_a denote the horizontal and vertical lines passing through a, respectively. Let $a, a' \in \mathbb{R}^2$ such that $a_x < a'_x$. If $a_y < a'_y$, then let

$$NW(a, a') = \{(x, y) \in S \mid x < a'_x, y > a_y\}, \qquad SE(a, a') = \{(x, y) \in S \mid x > a_x, y < a'_y\}.$$

Otherwise, if $a_y > a'_y$, let

$$NE(a, a') = \{(x, y) \in S \mid x > a_x, y > a'_y\}, \qquad SW(a, a') = \{(x, y) \in S \mid x > a'_x, y < a_y\}.$$

Suppose that we know the x-order of S, denoted S_x and computed in $O(n \log n)$ time. Let p and q be two elements of S such that q is below h_p, that is, $q_y < p_y$. Assume w.l.o.g. that $p_x < q_x$ (the other case, where $p_x > q_x$, is symmetric). We show now how to compute in $O(n)$ time the minimum of $d(S, \Pi)$ subject to p being the topmost point of S_3, and q, the bottommost point of S_1 (see Fig. 3(b)). Observe that this is equivalent to finding a pair of points $p', q' \in S$, as shown in Fig. 3(b), such that:

$$\Pi = \Big\{\big\{NE(q', q) \cup \{q\}\big\}, \big\{NW(p, q') \cup \{q'\}\big\}, \big\{SW(p, p') \cup \{p\}\big\}, \big\{SE(p', q) \cup \{p'\}\big\}\Big\},$$

is a rectangle partition of S, and $d(S, \Pi)$ is maximized, where

$$d(S, \Pi) = \min\Big\{\nabla\big(NE(q', q) \cup \{q\}\big), \nabla\big(NW(p, q') \cup \{q'\}\big), \nabla\big(SW(p, p') \cup \{p\}\big), \nabla\big(SE(p', q) \cup \{p'\}\big)\Big\}.$$

This can be done as follows:

We first assign a weight α to each element $u \in S_x$ so that

$$\alpha(u) = \begin{cases} \min\big\{\nabla(NE(u, q) \cup \{q\}), \nabla(NW(p, u) \cup \{u\})\big\} & u_x \in (p_x, q_x), u_y > p_y \\ -\infty & \text{otherwise.} \end{cases}$$

This assignment of weights can be done in $O(n)$ time. After that, we preprocess S_x in $O(n)$ time, by using a range maximum query data structure [3,10,17,18] so that, given any interval of contiguous elements of S_x, a point u of the interval maximizing $\alpha(u)$ can be reported in $O(1)$ time.

Let S'_x denote the subsequence of S_x composed of the points u such that $u_x > p_x$ and $u_y < q_y$, and $p_1, p_2, \dots, p_t$ denote the left to right ordering of the elements of S'_x. Observe that $W(SW(p, p_1) \cup \{p\})$ and $W(SE(p_1, q) \cup \{p_1\})$ can be computed in $O(n)$ time and that, for $j = 2, \dots, t$, both $W(SW(p, p_j) \cup \{p\})$ and $W(SE(p_j, q) \cup \{p_j\})$ can be computed in $O(1)$ time from $W(SW(p, p_{j-1}) \cup \{p\})$ and $W(SE(p_{j-1}, q) \cup \{p_{j-1}\})$. Then, in $O(n)$ time we can compute the values

$$\beta(p_j) = \min\Big\{\nabla(SW(p, p_j) \cup \{p\}), \nabla(SE(p_j, q) \cup \{p_j\})\Big\}$$

for all $j \in [1..t]$. Furthermore, in $O(n)$ time we can also compute for all $j \in [1..t]$ the point $u(p_j)$ located in the closed strip bounded by the lines h_p and h_q, and to the left of the line v_{p_j}.

For $j = 1, \dots, t$, we set $p' = p_j$ and compute the point q' with a constant-time range maximum query in S_x, that is, q' is a point of S_x located between the points p' and $u(p')$ that maximizes $\alpha(q')$. We then return the pair of points p' and q' that maximize $\min\{\beta(p'), \alpha(q')\}$.

Repeating the above $O(n)$-time algorithm for every two points $p, q \in S$ of S where q is below h_p, the result follows. □

4. Conclusions and open problems

In this paper, we showed that the coarseness of a 2-colored point set can be efficiently calculated for various cases in which the type of partitions allowed is constrained. When the point partition is given by parallel lines (in any direction) the problem can be solved in $O(n^2 \log^2 n)$ time and shown to be 3SUM-hard in two dimensions, and $O(n \log n)$ time in one dimension. When the clusters are given by pairwise-disjoint boxes and the number of boxes is fixed at k, we gave an $O(n^2 k)$-time algorithm in one dimension. For $k = 2, 3, 4$, $O(n \log n)$-time algorithms are given. We also proved NP-hardness in two dimensions, and presented algorithms running in $O(n \log n)$ time for $k = 2, 3$, and $O(n^3)$ time for $k = 4$. For a general fixed value of k, we showed how to compute the k-rectangle coarseness in $O(n^{2k-1})$ time. We also showed how to compute a constant-approximation to the k-rectangle coarseness in the general case where k is not constrained.

Various open problems remain, such as finding approximation algorithms for k-rectangle coarseness, and further studying the computational complexity of rectangle coarseness.

Acknowledgments

J.M. Díaz-Báñez was partially supported by project FEDER MEC MTM2009-08652, and MTM2016-76272-R AEI/FEDER,UE. C. Ochoa is supported by CONICYT-PCHA/Doctorado Nacional/ 2013-63130161 (Chile). P. Pérez-Lantero was partially supported by projects CONICYT FONDECYT/Iniciación 11110069 (Chile), CONICYT FONDECYT/Regular 1160543 (Chile), and Millennium Nucleus Information and Coordination in Networks ICM/FIC RC130003 (Chile).

References

[1] J.R. Alexander, J. Beck, W.W. Chen, 13 geometric discrepancy theory and uniform distribution, in: Handbook of Discrete and Computational Geometry, 2004, p. 279.
[2] C. Bautista-Santiago, J.M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, J. Urrutia, I. Ventura, Computing optimal islands, Oper. Res. Lett. 39 (4) (2011) 246–251.

[3] M.A. Bender, M. Farach-Colton, G. Pemmasani, S. Skiena, P. Sumazin, Lowest common ancestors in trees and directed acyclic graphs, J. Algorithms 57 (2) (2005) 75–94.
[4] S. Bereg, S. Cabello, J.M. Díaz-Báñez, P. Pérez-Lantero, C. Seara, I. Ventura, The class cover problem with boxes, Comput. Geom. 45 (7) (2012) 294–304.
[5] S. Bereg, J.M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, C. Seara, J. Urrutia, On the coarseness of bicolored point sets, Comput. Geom. 46 (1) (2013) 65–77.
[6] E. Boros, V. Gurvich, Y. Liu, Comparison of convex hulls and box hulls, Ars Combin. 77 (2005) 193–204.
[7] V. Capoyleas, G. Rote, G. Woeginger, Geometric clusterings, J. Algorithms 12 (2) (1991) 341–356.
[8] L.E. Caraballo, C. Ochoa, P. Pérez-Lantero, J. Rojas-Ledesma, Matching colored points with rectangles, J. Comb. Optim. (2015).
[9] B. Chazelle, The Discrepancy Method: Randomness and Complexity, Cambridge University Press, 2000.
[10] K.-Y. Chen, K.-M. Chao, On the range maximum-sum segment query problem, Discrete Appl. Math. 155 (16) (2007) 2043–2052.
[11] C. Cortés, J.M. Díaz-Báñez, P. Pérez-Lantero, C. Seara, J. Urrutia, I. Ventura, Bichromatic separability with two boxes: A general approach, J. Algorithms 64 (2–3) (2009) 79–88.
[12] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars, Computational Geometry: Algorithms and Applications, third ed., Springer-Verlag TELOS, Santa Clara, CA, USA, 2008.
[13] J.M. Díaz-Báñez, R. Fabila-Monroy, P. Pérez-Lantero, I. Ventura, New results on the coarseness of bicolored point sets, Inform. Process. Lett. 123 (2017) 1–7.
[14] D.P. Dobkin, D. Eppstein, D.P. Mitchell, Computing the discrepancy with applications to supersampling patterns, ACM Trans. Graph. 15 (4) (1996) 354–376.
[15] A. Dumitrescu, J. Pach, Partitioning colored point sets into monochromatic parts, Internat. J. Comput. Geom. Appl. 12 (05) (2002) 401–412.
[16] J. Eckstein, P.L. Hammer, Y. Liu, M. Nediak, B. Simeone, The maximum box problem and its application to data analysis, Comput. Optim. Appl. 23 (3) (2002) 285–298.
[17] H.N. Gabow, J.L. Bentley, R.E. Tarjan, Scaling and related techniques for geometry problems, in: Proc. of the 16th Annual ACM Symposium on Theory of Computing, ACM, 1984, pp. 135–143.
[18] D. Harel, R.E. Tarjan, Fast algorithms for finding nearest common ancestors, SIAM J. Comput. 13 (2) (1984) 338–355.
[19] A. Kaneko, M. Kano, Discrete geometry on red and blue points in the plane a survey, in: Discrete and Computational Geometry, Springer, 2003, pp. 551–570.
[20] S. Majumder, S.C. Nandy, B.B. Bhattacharya, Separating multi-color points on a plane with fewest axis-parallel lines, Fund. Inform. 99 (3) (2010) 315–324.
[21] C. Seara, On Geometric Separability (Ph.D. thesis), Univ. Politecnica de Catalunya, 2002.
[22] P. Serafini, Classifying negative and positive points by optimal box clustering, Discrete Appl. Math. 165 (2014) 270–282.
[23] P. Shirley, Discrepancy as a quality measure for sample distributions, in: Proc. Eurographics, Vol. 91, 1991, pp. 183–194.
[24] V. Spinelli, Supervised box clustering, in: Advances in Data Analysis and Classification, 2016, pp. 1–26.
[25] S.M. Weiss, C.A. Kulikowski, Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991.