

Smallest Color-Spanning Objects

Manuel Abellanas* Ferran Hurtado** Christian Icking* Rolf Klein**
Elmar Langetepe** Lihong Ma** Belén Palop** Vera Sacristán**

April 2001

Abstract

Motivated by questions in location planning, we show for a set of colored point sites in the plane how to compute the smallest (by perimeter or area) axis-parallel rectangle, the narrowest strip, and other smallest objects enclosing at least one site of each color.

1 Introduction

We are given a set of n point sites in the plane and $k \leq n$ colors, each site is associated one color. A region of the plane is called *color-spanning* if it contains at least one point of each color. For different kinds of regions we are interested in the smallest color-spanning one of that kind.

The original motivation for our questions comes from location planning. Suppose there are k types of facilities, e. g. schools, post offices, supermarkets, modeled by n colored points in the plane, each type by its own color. One basic goal in choosing a residence location is in having at least one representative of each facility type in the neighborhood, where there are various specifications of the term “neighborhood”. A natural question is to ask for the smallest color-spanning *circle*. It can be found using the upper envelope of *Voronoi surfaces*,

* Dept. de Matemática Aplicada, Universidad Politécnica de Madrid, Spain

** Dept. de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Barcelona, Spain

* Praktische Informatik VI, FernUniversität Hagen, Germany

** Institut für Informatik I, Universität Bonn, Germany

** ESCET, Universidad Rey Juan Carlos, Madrid, Spain

as described by Huttenlocher et al. [HKS93] and Sharir and Aggarwal [SA95, Section 8.7]; their algorithm for computing the solution runs in time $O(kn \log n)$. Similarly one can determine the smallest color-spanning axis-parallel *square* and other objects with fixed orientation which are unit circles of a convex distance function.

In this paper, we propose to solve more complicated problems of this context: we give algorithms to compute the smallest color-spanning axis-parallel rectangle in Section 2 and the narrowest color-spanning strip in Section 3.

A couple of related optimization problems for a set S of n points have already been studied in the literature, with motivations from statistical clustering or pattern recognition. For example, the convex polygon with minimum perimeter containing k points of S can be found by using the methods of Dobkin et al. [DDG83], Aggarwal et al. [AIKS91], or finally Eppstein and Erickson [EE94], the last one in time $O(n \log n + k^3 n)$. The minimum *area* convex polygon containing k points of S can be determined in time $O(n^2 \log n + kn^2 \min(k^2, n))$ combining results of [EE94] and Eppstein et al. [EORW92]. Similar problems for selecting k points out of n use as optimization criterion the diameter or the variance of the k -set, or they ask for the smallest circle with respect to a certain metric containing at least k points [AIKS91, DDG83, ESZ94, EORW92, Mat95, Smi92], this latter problem is of course very closely related to the Voronoi diagram of order k .

Other very natural optimization criteria are the perimeter and the area of the axis-parallel rectangle enclosing a k -point set, these criteria are sometimes briefly called the L_∞ perimeter and L_∞ area. For computing the smallest perimeter, the best known running time is $O(n \log n + k^2 n)$ for algorithms by Datta et al. [DLSS95] and by Eppstein and Erickson [EE94]. The algorithm of Aggarwal et al. [AIKS91] can be used for both variants of the problem, the area and the perimeter, and takes time $O(\min(k^2 n \log n, n^3))$, while Segal and Kedem's solution [SK98] for both variants runs in $O(n + k(n - k)^2)$ time and is applicable only for $k > n/2$.

Not much seems to be known about lower bounds for these problems. Matoušek [Mat94] reports that at least some of them are known to be n^2 -hard, a notion introduced by Gajentaan and Overmars [GO95]; compare also Erickson and Seidel [ES95].

Interestingly, the approach in [AIKS91] for the smallest rectangle, like some approaches for the smallest circle, is also based on the Voronoi diagram of higher order, in this case of order $6k - 6$. This is because the optimal k -point set can be shown to be contained in a circle centered at such a Voronoi vertex which

passes through the corresponding sites. Eppstein and Erickson’s approach [EE94] uses the fact that the members of the optimal k -point set are always among the $16k$ nearest rectilinear neighbors of each of them. But neither of these properties based on proximity seem to be extensible to our new problem that involves colors.

For multicolored point sets, there are solutions to several problems, such as the *bichromatic closest pair*, see e.g. Preparata and Shamos [PS85, Section 5.7], Agarwal et al. [AESW91], and Graf and Hinrichs [GH93], the *group Steiner tree* where, for a graph with colored vertices, the objective is to find a minimum-weight subtree that covers all colors, see Mitchell [Mit00, Section 7.1], or the *chromatic nearest neighbor search*, see Mount et al. [MNSW00].

2 The smallest color-spanning rectangle

Among all rectangles whose sides are parallel to the x - and y -axis and which contain at least one site of each color we are looking for the smallest one, by perimeter or by area.

Some special cases are immediately solved. For $k = 1$ the problem is trivial, and for $k = n$, i. e., we have exactly one point for each color, the solution is the bounding box of the point set. And also the case $k = n - C$ for some constant C can be solved in time $O(n)$ because in this case there is only a constant ($\leq 2C$) number of sites with colors that have more than one site. Finally for $k = 2$ in the perimeter case, we can make use of an algorithm that computes the bichromatic L_1 -closest pair in time $O(n \log n)$, see e.g. Graf and Hinrichs [GH93]. For the general case however, new ideas are necessary.

In Section 2.1 we show that the optimal rectangle must fulfill the so-called *non-shrinkable* property, we present a simple algorithm with running time in $O(n(n - k)^2)$, and we prove the tight bound of $\Theta((n - k)^2)$ for the number of non-shrinkable rectangles. In Section 2.2 we give necessary and sufficient conditions for non-shrinkable rectangles, and we refine the algorithm to a running time of $O(nk(n - k))$. Finally, in Section 2.3 we use a result by Overmars and van Leeuwen [OvL81] for dynamically maintaining the maximal elements of a point set to further improve the running time to a near-optimal $O(n(n - k) \log^2 k)$.

2.1 Non-shrinkable rectangles and a first algorithm

Let p_x and p_y denote the coordinates of a site and p_{col} denote its color. For the sake of simplicity of the presentation, we make the following assumption on general

position. No two x - or y -coordinates are equal, i. e., there is no horizontal or vertical line passing through two points. We exclude the trivial cases and assume for the remaining part of the paper that $1 < k < n$. It is clear that the smallest color-spanning rectangle, by perimeter or by area, must be non-shrinkable in the following sense.

Definition 1 An axis-parallel rectangle is called *non-shrinkable* if it contains sites of all k colors and it does not properly contain another axis-parallel rectangle that contains all colors.

Therefore, each non-shrinkable rectangle must touch a site with each of its four edges, such that there are two, three, or four sites on its boundary, among them no two of the same color. The colors on its boundary do not appear at sites in its interior.

Our algorithms will systematically find all non-shrinkable rectangles and compare their perimeters or areas to determine the smallest one, thereby solving the two variants of the problem at the same time. A first and quite simple idea to do this is shown next, this is similar to the procedure of [AIKS91].

Algorithm 1 The lower-left corner of a candidate is either determined by one site or by a pair of sites of different colors such that these two sites lie on the candidate's bottom and left edges.

For each such lower-left corner, we proceed as follows. Let U be the set of sites which lie above and to the right of the corner.

1. Initially the top edge of the rectangle starts at infinity. The right edge starts at the x -coordinate of the corner, it is moved right over the sites of U until all colors are contained in the actual rectangle.
2. Then, in decreasing y -order, the top edge steps through the sites of the rectangle as long as it still contains all colors; when this stops, we have found a candidate.
3. The right edge advances until it reaches a site with the color of the site at the actual top edge.
4. As long as the right edge has not stepped over all points of U , we repeat from step 2.

It is clear that all non-shrinkable rectangles are checked as candidates by Algorithm 1, but also some more rectangles that may contain sites of the same

color as the left or bottom edges. For each corner the algorithm spends time $O(n)$ if the sites are previously sorted by their x -coordinates and by their y -coordinates.

Remark that for one fixed corner we cannot have more than $n-k+1$ candidates because the right edge has stepped over at least k sites in step 1. Furthermore, the left edge of a candidate can be only at the first $n-k+1$ sites in x -order and the lower edge only at the first $n-k+1$ sites in y -order, so we obtain a $O(n(n-k)^2)$ bound for the running time of Algorithm 1.

Before trying to improve on this time bound, we are interested in determining the exact number of non-shrinkable rectangles, since in the worst case it seems unavoidable to check (nearly) all of them.

Lemma 2 *There are $\Theta((n-k)^2)$ non-shrinkable rectangles.*

Proof. We start by proving the upper bound. As we have remarked earlier, each edge of a non-shrinkable rectangle N must contain a site of a color that occurs only once in N . First consider the case that a site is a corner of the rectangle, i. e., the site touches two edges. A site can be the, e. g., lower left corner of a non-shrinkable rectangle only if it belongs to the $n-k+1$ leftmost sites because it must have $k-1$ sites to its right. Also, in the analysis of Algorithm 1 we have seen that there are at most $n-k+1$ non-shrinkable rectangles for one fixed corner. Thus, the upper bound holds for all non-shrinkable rectangles that have at least one site at a corner. In particular, this also settles the cases $k=2, 3$.

So we can assume that $k \geq 4$ and that each edge contains exactly one site in its interior. Let l , b , and r denote the colors of the singular points on the left, bottom, and right edge of N , correspondingly. We enlarge N by moving its upper edge upwards until one of the following events occurs. Either, the upper edge hits a point of color b ; then we have obtained a so-called *enlarged candidate* with singular points on its left and on its right edge that contains points of color b on its top and bottom edges, but no further b -colored points (type 1). Or the upper edge hits a point of color l or r , say l ; then we have an enlarged candidate containing two points of color l on its top and left edges, but no further points of color l , and singular points on its right and bottom edges (type 2). If the upper edge does not hit a point of color b , l , or r then we obtain an enlarged candidate with upper edge at infinity and singular points on its other three edges (type 3).

This way we have mapped each non-shrinkable rectangle on an enlarged candidate of type 1, 2, or 3. The mapping is one-to-one; given an enlarged candidate of any type we can just lower its top edge until, for the first time, the lowest point

of some color is hit, and obtain a non-shrinkable rectangle. Thus, it suffices to show that there are only $O((n - k)^2)$ enlarged candidates of each type.

In order to bound the number of type 1 rectangles we fix an arbitrary point, p , of color p_{col} and show that there are at most $O(n - k)$ type 1 rectangles $R_{i,j}$ that have p on their bottom edge and another p_{col} -colored point q_i on their top edge to the right of p . Indexing is such that q_1, \dots, q_m have increasing x -coordinates. Let $r_{i,j}$ be the singular point on the right edge of rectangle $R_{i,j}$, for $1 \leq j \leq m_i$. Clearly, different rectangles $R_{i,j}$ with the same index i must have different right points $r_{i,j}$. Since none of these rectangles can contain a third point of color p_{col} , point q_{i+1} must be below q_i and to the right of all points $r_{i,j}$, $1 \leq j \leq m_i$. Trivially, all points $r_{i+1,j}$, $1 \leq j \leq m_{i+1}$, are to the right of q_{i+1} . A continuation of this argument shows that *all* points $r_{i,j}$ are pairwise different. This proves the claim since only $n - k + 1$ points can have the right edge, and the same for the bottom edge.

The argument for the type 2 rectangles is quite similar. We fix the point p on the left edge and consider all rectangles $R_{i,j}$ of type 2 that have a point q_i of the same color on their top edges. Again, all singular points $r_{i,j}$ on the right edges are pairwise different.

The unbounded enlarged candidates of type 3 are even easier to count: for a fixed point on the bottom edge there can be only $n - k + 1$ of them, since for each possible left edge there is at most one right edge, if any, and only $n - k + 1$ sites can have the left edge. (By the way, k is also an upper bound on the number of rectangles of this type for a fixed bottom edge, as will follow from Lemma 3.)

It remains to show the lower bound, i. e., we are given numbers n and k , and we want to place n sites with k colors such that there are $\Omega((n - k)^2)$ different non-shrinkable rectangles. To this end, we make a construction as sketched in Figure 1.

We construct three groups of sites. The first group consists of $\lfloor (n - k)/2 \rfloor + 1$ sites of color 1 and is placed on the line $y = -1 - x$ at positions with negative x - and y -coordinates, the second group has $\lceil (n - k)/2 \rceil + 1$ sites but of color 2 and is placed on the line $y = 1 - x$ at positions with positive coordinates, and the third group contains one site of each of the other $k - 2$ colors and is placed very close to the origin.

Now each rectangle spanned by a site of color 1 as the lower left corner and by a site of color 2 as the upper right corner contains all colors and is one of $\Omega((n - k)^2)$ non-shrinkable rectangles. \square

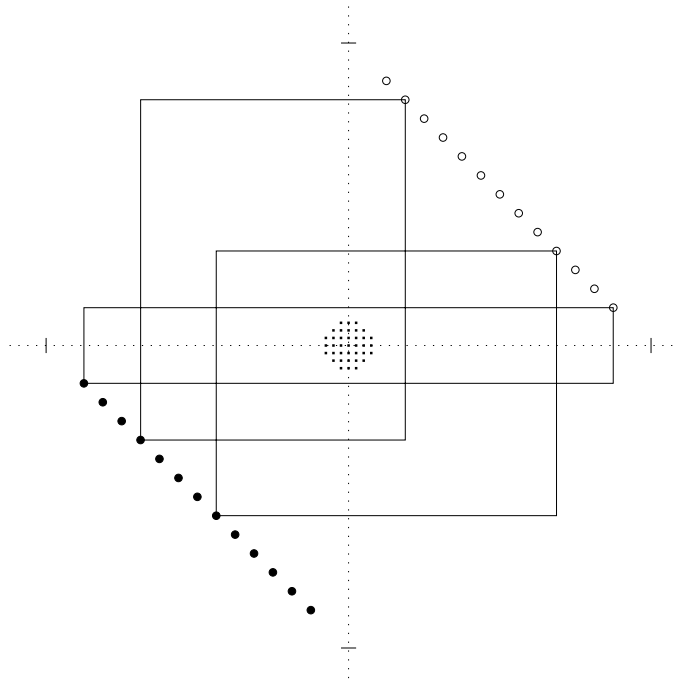


Figure 1: Each rectangle spanned by a site of color 1 and a site of color 2 is non-shrinkable.

2.2 An improved approach

The question arises if the proof method for the $O((n - k)^2)$ upper bound can be used for efficiently constructing all non-shrinkable rectangles. In fact, we are able to enumerate all enlarged candidates of types 1, 2, and 3 within time $O(n^2 \log k)$. The difficulty is in efficiently moving down the upper edges of these rectangles, in order to obtain non-shrinkable rectangles. This can be done within the same time bound for the types 2 and 3, but seems quite hard to do for the type 1 rectangles.

Therefore, we resort to a more direct method that is a refinement of Algorithm 1. Instead of fixing the lower left corner, let us try to fix the upper and lower edges, i. e., for each pair of sites a and b with $a_y < b_y$ we check all non-shrinkable rectangles with lower y -coordinate a_y and upper y -coordinate b_y .

We consider conditions that must be fulfilled by such a non-shrinkable rectangle with left edge at l and right edge at r , l and r may coincide with a or b . First, it is clear that a and b must be contained in the rectangle. Second, the interior of the rectangle must not contain sites of the colors of a and b . Third, the colors of l and r are not contained in the interior either.

More formally, for a given color c we define the following numbers.

$$L_c(a, b) = \max_{p \in S, p_{col}=c} \{ p_x \mid a_y < p_y < b_y \text{ and } p_x < a_x \}$$

$$R_c(a, b) = \min_{p \in S, p_{col}=c} \{ p_x \mid a_y < p_y < b_y \text{ and } p_x > a_x \}$$

In other words, $L_c(a, b)$ is the maximum x -coordinate of all sites of color c in the horizontal strip between a and b and to the left of a_x , and $R_c(a, b)$ the analogous minimum to the right of a_x ; they take on the values of $-\infty$ resp. $+\infty$ if no such site exists.

Now the first condition above means that

$$l_x \leq \min(a_x, b_x) \quad \text{and} \quad r_x \geq \max(a_x, b_x). \quad (1)$$

The second condition can be expressed as

$$l_x > \max(L_{a_{col}}(a, b), L_{b_{col}}(a, b)) \quad \text{and} \quad r_x < \min(R_{a_{col}}(a, b), R_{b_{col}}(a, b)). \quad (2)$$

In other words, we have an x -interval for the possible positions of the left edge of a non-shrinkable rectangle from a_y to b_y , and another one for the right edge.

The third condition transforms to

$$l_x = L_{l_{col}}(a, b) \text{ if } l \neq a, b \quad \text{and} \quad r_x = R_{r_{col}}(a, b) \text{ if } r \neq a, b, \quad (3)$$

i. e., the site l on the left edge, if it is not a or b itself, is the x -maximal site of its color in the horizontal strip between a and b and to the left of $\min(a_x, b_x)$, and correspondingly with r . Therefore the following assertion holds.

Lemma 3 *Let a and b be two sites of S . Independently of n , there are at most $k - 1$ non-shrinkable rectangles with lower edge at a and upper edge at b .*

Proof. According to (3), the left edge of such a non-shrinkable rectangle has only $k - 2$ possible positions if its color is different from a_{col} and b_{col} , and $\min(a_x, b_x)$ as one additional possibility. \square

For fixed a , the quantities $L_c(a, b)$ and $R_c(a, b)$ can easily be updated if b steps through the sites in y -order. For each b it remains to match the correct pairs of sites at the left and at the right edges, this is done by the following algorithm.

Algorithm 2 The sites are sorted in y -order. For each site a we do the following to find all candidate rectangles with lower y -coordinate a_y .

1. Let L and R be arrays over all colors, initialized to $-\infty$ resp. $+\infty$; they will contain the values $L_c(a, b)$ and $R_c(a, b)$ for the actual a and b and for all colors c .

The lists $SortL$ and $SortR$ will contain all sites that actually contribute an entry to L resp. R , sorted in x -direction.

2. For all sites b with $b_y > a_y$ in y -order we do. Perform steps **2a** to **2c** only if $b_{col} \neq a_{col}$, in any case perform step **2d**.

- (a) $InclL := \min(a_x, b_x)$; $ExclL := \max(L_{a_{col}}, L_{b_{col}})$;
 $InclR := \max(a_x, b_x)$; $ExclR := \min(R_{a_{col}}, R_{b_{col}})$;

In list $SortL$ we mark the sites with x -coordinates greater than $ExclL$ and smaller than $InclL$, and correspondingly with $SortR$ from $InclR$ to $ExclR$.

- (b) The left edge starts at the first marked element of $SortL$. The right edge starts at $InclR$ and if necessary steps over the marked sites in $SortR$ until all colors are contained in the actual rectangle.
- (c) As long as the marked parts of $SortL$ and $SortR$ are not exhausted, we repeat the following steps.
 - i. The left edge advances over the marked sites in $SortL$ and finally $InclL$ as long as the rectangle contains all colors; when this stops, we have found a candidate.
 - ii. The right edge advances over the marked sites in $SortR$ until it reaches the color of the site at the actual left edge.
- (d) If $b_x < a_x$
then $L_{b_{col}} := \max(L_{b_{col}}, b_x)$, also unmark and update $SortL$
else $R_{b_{col}} := \min(R_{b_{col}}, b_x)$, also unmark and update $SortR$.

Lemma 4 *The candidates reported by Algorithm 2 are precisely all non-shrinkable rectangles. Its running time is in $O(nk(n - k))$.*

Proof. Let us consider what happens if steps **2a** to **2d** are executed for certain sites a and b .

Step **2d** has been performed for all previous values of b , so L and R contain the correct $L_c(a, b)$ and $R_c(a, b)$ for all colors. Remark that this also holds for b_{col} because the update of L and R concerning b is done at the end of the loop.

$SortL$ and $SortR$ contain the sites corresponding to the values of L resp. R , and only these values are possible left resp. right edges of the rectangle, as we have seen earlier. The marked parts correspond to the intervals $(ExclL, InclL)$ resp. $(InclR, ExclR)$ and reflect conditions (1) and (2); sites a or b can also be at the left or right edges of a non-shrinkable rectangle, this is taken into account by starting the right edge at $InclR$ in step 2b and finishing with the left edge at $InclL$ in step 2(c)i.

For each possible left edge the matching right edge, if any, is found in steps 2(c)i and 2(c)ii, these steps are very similar to steps 2 to 3 of Algorithm 1.

The case in which there is no non-shrinkable rectangle for a at the bottom edge and b at the top is quickly detected: If some colors are missing in the horizontal strip between a and b then step 2b already does not succeed. If another site of color a_{col} or b_{col} is contained in the rectangle spanned by a and b then $ExclL > InclL$ or $InclR > ExclR$ and one of the lists is not marked at all. Finally, the case $b_{col} = a_{col}$ is explicitly excluded in step 2.

The running time can be estimated as follows. Site a at the bottom edge needs to be iterated only over the first $n - k + 1$ sites in y -order, so this factor is contributed. Factor n is for the loop over all b (not $n - k$ because the updates in step 2d need to be executed for all b above a). Finally, the repetition of steps 2(c)i and 2(c)ii results in a factor k , as well as the (un)marking and the updates of the sorted lists. \square

For small and in particular for constant k Algorithm 2 is the method of choice, because it is very simple and can be implemented using just a few lines of code. On the other hand, for large k the $O(n(n - k)^2)$ method of Algorithm 1 is preferable. In the general case however, there is still room for improvements.

2.3 Maximal elements

Definition 5 A *maximal element* of a set, T , of points in the plane is a point $p \in T$ such that there is no $q \in T$ with $p_x > q_x$ and $p_y < q_y$.

Remark that for our special purpose we have slightly deviated from the usual definition, see [OvL81], we are interested in maximal elements in the upper *left* (instead of right) direction, see Figure 2. Our maximal elements are those that are not *dominated* from left and above by another point of the set.

Now consider, for given a and b , the values of $L_c(a, b)$ and $R_c(a, b)$. We transform these values to points in 2D, using L for the x -coordinates and R for

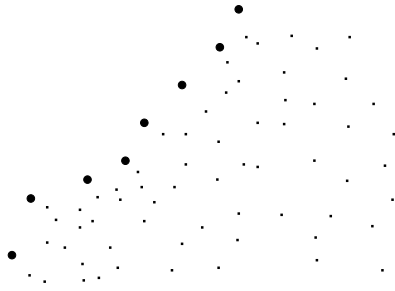


Figure 2: Maximal elements of a point set in the upper left direction.

the y -coordinates.

$$T_c(a, b) = (L_c(a, b), R_c(a, b)) \text{ for all colors } c \neq a_{col}, b_{col}$$

Some of the coordinates of these points may be $\pm\infty$. With $T(a, b)$ we denote the set of all points $T_c(a, b)$. The next lemma shows that maximal elements are closely related to spanning colors.

Lemma 6 *Assume that the horizontal strip between a and b contains all colors. The point $T_c(a, b)$ for some color c is a maximal element of $T(a, b)$ if and only if the rectangle with a and b at the bottom and top edges and with $L_c(a, b)$ as left and $R_c(a, b)$ as right edge contains all colors with the possible exception of b_{col} .*

Proof. Let $T_c(a, b)$ be a maximal element of $T(a, b)$. Suppose there is a color, c' , which is not contained in the rectangle between a , b , $L_c(a, b)$, and $R_c(a, b)$. Then $L_{c'}(a, b) < L_c(a, b)$ and $R_{c'}(a, b) > R_c(a, b)$, and $T_{c'}$ dominates $T_c(a, b)$, a contradiction. Conversely, if all colors are contained in the rectangle then $T_c(a, b)$ must be a maximal element because it can't be dominated by any other color. \square

Now we have an interesting relation between non-shrinkable rectangles and maximal elements.

Lemma 7 *For a non-shrinkable rectangle with sites a, b, l, r at the bottom, top, left, and right edges with $l \neq a, b$ and $r \neq a, b$, $T_{l_{col}}(a, b)$ and $T_{r_{col}}(a, b)$ are two successive maximal elements of the set of points in $T(a, b)$.*

Proof. Assume that $T_{l_{col}}(a, b)$ is dominated by some $T_c(a, b)$. Clearly we have $L_c(a, b) < L_{l_{col}}(a, b) = l_x$, but also $R_c(a, b) > R_{l_{col}}(a, b) > r_x$ holds because l_{col} cannot appear a second time in the rectangle. This means that color c is not contained in the rectangle, a contradiction, and analogously for $T_{r_{col}}(a, b)$.

Now assume some $T_c(a, b)$ is maximal element between the two maximal elements $T_{r_{col}}(a, b)$ and $T_{l_{col}}(a, b)$. Then we have $L_{r_{col}}(a, b) < L_c(a, b) < L_{l_{col}}(a, b) = l_x$ and $r_x = R_{r_{col}}(a, b) < R_c(a, b) < R_{l_{col}}(a, b)$, and again c is not contained in the rectangle. \square

And the converse is also true, in some sense.

Lemma 8 *Consider two sites a and b and two colors $c, c' \neq a_{col}, b_{col}$ such that $T_c(a, b)$ and $T_{c'}(a, b)$ are two successive maximal elements of $T(a, b)$ and assume that the horizontal strip between a and b contains all colors. Then the rectangle between a, b, l with $l_x = L_{c'}(a, b)$ as left edge, and r with $r_x = R_c(a, b)$ as right edge is non-shrinkable if additionally conditions (1) and (2) hold.*

Proof. From Lemma 6 we know that the rectangle between $a, b, L_c(a, b)$ and $R_c(a, b)$ contains all colors. Now let the left edge move right until the rectangle is non-shrinkable. The left edge must now be situated at $L_{c'}(a, b)$, otherwise there would be another maximal element between $T_c(a, b)$ and $T_{c'}(a, b)$. Conditions (1) and (2) are necessary to guarantee that no other sites of color a_{col} or b_{col} are contained in the rectangle. \square

Theorem 9 *Given n sites and k colors, the smallest color-spanning rectangle can be found in time $O(n(n - k) \log^2 k)$.*

Proof. We modify Algorithm 2. The main difference is in maintaining a dynamic tree *MaxElem* of maximal elements instead of the lists *SortL* and *SortR*.

In step 2d now *MaxElem* is updated if the value of $L_{b_{col}}$ or $R_{b_{col}}$ has changed; this can be done in time $O(\log^2 k)$ using the method of Overmars and van Leeuwen [OvL81].

The marking of the lists is replaced in the following way. The values *ExclL*, *InclL*, *InclR*, and *ExclR* are computed as before. Then the subsequence of elements in *MaxElem* is extracted that is included in $(ExclL, InclL)$ in x -direction as well as in $(InclR, ExclR)$ in y -direction. This can be done in time $O(\log k)$ plus the length of this subsequence which in turn is essentially the same as the number of non-shrinkable rectangles reported. It remains to report the matchings between left and right edges as described in Lemma 8.

So the running time of this method is $O(n(n - k) \log^2 k)$ plus the total number of reported non-shrinkable rectangles but which is fortunately bounded by $O((n - k)^2)$, see Lemma 2. \square

3 The narrowest color-spanning strip

The narrowest color-spanning strip problem asks for two parallel lines in the plane that contain all colors in between them such that their distance is minimized.

Notice that the solution strip must have three sites of three different colors on its boundary, because if they were only two or they had a coincident color, the strip could be shrunk by rotation.

A brute force approach could work as follows. Consider the $O(n^2)$ lines defined by two sites of different colors, and sort them by slopes in $O(n^2 \log n)$ time. Start with one of them and project all the other sites, following the direction of the line, onto any perpendicular line. By sorting the projected points and walking along them we can find the solution in that direction in $O(n \log n)$ time. Now, at each change of direction, we only need to update the order of the projected points in $O(1)$ time and explore the points, again walking along them, in $O(n)$ time to find the solution for the new direction. Hence, the algorithm works in $O(n^3)$ time.

When the direction changes, the cluster of points that gives the optimal solution may completely change. This is the reason why we don't envisage a more clever updating. Using techniques of inversion and outer envelopes we obtain a much better algorithm, as the following theorem states.

Theorem 10 *Given n sites and k colors, the narrowest color-spanning strip can be found in $O(n^2 \alpha(k) \log k)$ time.*

Proof. Consider a site b of a given color. For each other color site r , the strips that go through b and r have the following property: the orthogonal projection of b onto the lines through r (i. e. the point that determines the width of the strip) describes a circle that has br as diameter, see Figure 3.

If we fix the point b and consider all the circles corresponding to a given different color, say red, we obtain an arrangement of (red) circles, all passing through b . The width of a strip containing b and exactly one red site is given by the inner envelope of the arrangement. After an inversion with respect to the point b , the red circles get transformed into red lines, and the distances from b are inverted. Hence, the width of a strip containing b and exactly one red site is given by the outer envelope of the arrangement of lines. As each two lines intersect at most once, the complexity of such envelope is $O(n_i)$, and it can be computed in $O(n_i \log n_i)$ time (for example, by divide and conquer, because the

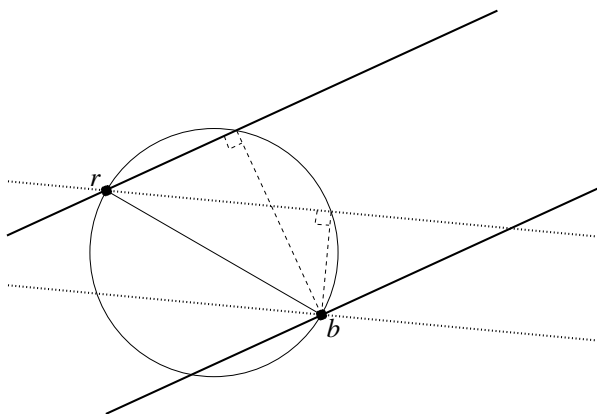


Figure 3: The locus of the orthogonal projection of b onto the lines through r is the circle that has br as diameter.

merge can be done in time linear to the result, as we know that the point b is lying in the kernel of the starshaped polygons to be merged).

Now, for each site, all the outer envelopes of all the different colors can be computed in $O(n \log n)$ time. The one-of-a-set spanning strips are determined by the inner envelope of the arrangement of the 1-colored outer envelopes. This envelope has complexity $O(n\alpha(k))$ and can be computed in $O(n\alpha(k) \log k)$ time [SA95, Corollary 6.3].

As this is done for each site, the algorithm runs in $O(n^2\alpha(k) \log k)$ time. \square

4 Conclusions

We have solved two optimization problems by giving algorithms that are likely to be close to optimal. The narrowest color-spanning strip problem can be solved in time $O(n^2\alpha(k) \log k)$, and it is n^2 -hard in the sense of [GO95], this proof is not very difficult but omitted here for brevity.

On the other hand, the smallest color-spanning rectangle problem can be solved in time $O(n(n-k) \log^2 k)$ while we have the tight quadratic bound for the number of non-shrinkable rectangles. It would be interesting to have a formal proof (or a refutation) that this number is also a lower bound for this problem.

The smallest color-spanning rectangle with an arbitrary orientation would be the next natural generalization.

References

- [AESW91] Pankaj K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and Emo Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete Comput. Geom.*, 6(5):407–422, 1991.
- [AIKS91] A. Aggarwal, H. Imai, N. Katoh, and Subhash Suri. Finding k points with minimum diameter and related problems. *J. Algorithms*, 12:38–56, 1991.
- [DDG83] David P. Dobkin, Robert L. Drysdale, III, and Leonidas J. Guibas. Finding smallest polygons. In Franco P. Preparata, editor, *Computational Geometry*, volume 1 of *Adv. Comput. Res.*, pages 181–214. JAI Press, Greenwich, Conn., 1983.
- [DLSS95] A. Datta, H.-P. Lenhof, C. Schwarz, and M. Smid. Static and dynamic algorithms for k -point clustering problems. *J. Algorithms*, 19:474–503, 1995.
- [EE94] D. Eppstein and J. Erickson. Iterated nearest neighbors and finding minimal polytopes. *Discrete Comput. Geom.*, 11:321–350, 1994.
- [EORW92] D. Eppstein, M. H. Overmars, Günter Rote, and G. Woeginger. Finding minimum area k -gons. *Discrete Comput. Geom.*, 7:45–58, 1992.
- [ES95] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Comput. Geom.*, 13:41–57, 1995.
- [ESZ94] A. Efrat, Micha Sharir, and A. Ziv. Computing the smallest k -enclosing circle and related problems. *Comput. Geom. Theory Appl.*, 4:119–136, 1994.
- [GH93] T. Graf and K. Hinrichs. Algorithms for proximity problems on colored point sets. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 420–425, 1993.
- [GO95] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom. Theory Appl.*, 5:165–185, 1995.
- [HKS93] D. P. Huttenlocher, K. Kedem, and Micha Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete Comput. Geom.*, 9:267–291, 1993.

- [Mat94] J. Matoušek. On geometric optimization with few violated constraints. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 312–321, 1994.
- [Mat95] J. Matoušek. On enclosing k points by a circle. *Inform. Process. Lett.*, 53:217–221, 1995.
- [Mit00] Joseph S. B. Mitchell. Geometric shortest paths and network optimization. In Jörg-Rüdiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, pages 633–701. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.
- [MNSW00] David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. Chromatic nearest neighbour searching: a query sensitive approach. *Comput. Geom. Theory Appl.*, 17:97–119, 2000.
- [OvL81] M. H. Overmars and J. van Leeuwen. Maintenance of configurations in the plane. *J. Comput. Syst. Sci.*, 23:166–204, 1981.
- [PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.
- [SA95] Micha Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, New York, 1995.
- [SK98] M. Segal and K. Kedem. Enclosing k points in the smallest axis parallel rectangle. *Inform. Process. Lett.*, 65:95–99, 1998.
- [Smi92] M. Smid. Finding k points with a smallest enclosing square. Report MPI-I-92-152, Max-Planck-Institut Inform., Saarbrücken, Germany, 1992.