Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik I

Manuel Abellanas[1] **Ferran Hurtado**[2]
Christian Icking[3] **Rolf Klein**[4]
Elmar Langetepe[4] **Lihong Ma**[4]
Belén Palop[5] **Vera Sacristán**[2]

The Farthest Color Voronoi Diagram and Related Problems

Technical Report 002
2006

[1] Dept. de Matemática Aplicada, Universidad Politécnica de Madrid, Spain
[2] Dept. de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Barcelona, Spain
[3] Prakt. Informatik VI, FernUniversität Hagen, Germany
[4] Institut für Informatik I, Universität Bonn, Germany
[5] Universidad de Valladolid, Spain

The Spanish authors acknowledge partial support from Acción integrada HA1999-0094, MEC-DGES-SEUID PB98-0933, and Gen. Cat. 1999SGR000356, while the German team was supported by DAAD grant 314-AI-e-dr.

Abstract

Given n point sites in the plane each painted in one of k colors, the region of a c-colored site p in the *Farthest Color Voronoi Diagram* (*FCVD*) contains all points of the plane for which c is the farthest color and p the nearest c-colored site. This novel structure generalizes both the standard Voronoi diagram ($k = 1$) and the farthest site Voronoi diagram ($k = n$). We show that the *FCVD* has structural complexity $\Theta(nk)$ if $k \leq \frac{n}{2}$, and we provide an $O(n^2\alpha(k)\log k)$ algorithm for computing the *FCVD*.

We call a set *color-spanning* if it contains at least one point of each color. From the *FCVD*, we can quickly determine the smallest color-spanning circle. Moreover, we present algorithms for computing a color-spanning axis parallel rectangle of minimum area or perimeter in time $O(n(n-k)\log^2 k)$ and for finding the narrowest color-spanning strip in time $O(n^2\alpha(k)\log k)$.

Keywords. Location planning, Voronoi diagram, smallest color-spanning circle, minimum color-spanning rectangle, narrowest color-spanning strip.

1 Introduction

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. In this paper we provide algorithms that may help to achieve this goal for various specifications of the term "neighborhood". Several problems on multicolored point sets have been previously considered, such as the *bichromatic closest pair*, see e. g. Preparata and Shamos [12, Section 5.7], Agarwal et al. [1], and Graf and Hinrichs [8], or the *group Steiner tree*, see Mitchell [10, Section 7.1].

Let us call a set *color-spanning* if it contains at least one point of each color. A natural approach to the above location problem is to ask for the center of the smallest color-spanning circle. For $k = n$ this amounts to finding the smallest circle enclosing n given points. This problem can be solved in time $O(n \log n)$ by means of the farthest site Voronoi diagram [3], in time $O(n)$ using Megiddo's linear programming method [9], or in randomized time $O(n)$ by Welzl's minidisk algorithm [14]. A special case is $k = 2$, then the solution is given by the bichromatic closest pair, see above. For $2 < k < n$ we know of no previous results. A brute force approach—check every circle defined by three sites for all colors—would take $O(n^4)$ time.

In order to find a better solution, we generalize, in Section 2, Voronoi

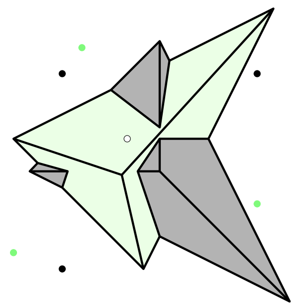

Figure 1: The farthest color Voronoi diagram for three colors and seven sites. The doubly connected unbounded area belongs to the ○ site in the middle while the dark (■) and light (■) shaded areas belong to the closest ● resp. ● sites.

diagrams in the following way. If p denotes a site of color c, we put all points of the plane in the region of p for which c is the farthest color, and p the nearest c-colored site, i. e., z belongs to the region of p iff the closed circle centered at z that passes through p contains at least one point of each color, but no point of color c is contained in its interior. We call the resulting planar subdivision the *Farthest Color Voronoi Diagram*, *FCVD* for short, see Figure 1 for an example. For $k = n$ the *FCVD* becomes the farthest site Voronoi diagram, while for $k = 1$ we obtain the standard Voronoi diagram of n points.

We show that the *FCVD* is of complexity $\Theta(nk)$ if $k \leq \frac{n}{2}$. We provide an algorithm for constructing the *FCVD* that runs in time $O(n^2\alpha(k)\log k)$. Once the *FCVD* is given, we can, in time $O(nk)$, determine the smallest color-spanning circle: its center is either a 3-colored vertex, or the midpoint of a 2-colored edge. We complement this result by a simple $O(k^3 n \log n)$ algorithm that runs faster for small values of k.

Clearly, the *FCVD* can be generalized to other distance measures, e. g. to convex distance functions. This would allow us to compute the smallest color-spanning rectangle of fixed orientation and fixed aspect ratio by the same method as before. An interesting problem arises if the aspect ratio is not specified.

So let us assume that, in the residence location problem, we want to determine a color-spanning axis parallel rectangle of minimum area or minimum perimeter. In principle, we could use a 3-dimensional *FCVD* of colored vertical lines for this problem, where a horizontal cross-section at height z equals the *FCVD* for aspect ratio z. However, we present a more direct approach in Section 3. Our algorithm constructs the smallest color-spanning rectangle in time $O(n(n-k)\log^2 k)$, using a technique by Overmars and van Leeuwen [11] for dynamically maintainig maximal elements. Again, we complement this result by simple algorithms whose $O(n(n-k)^2)$ resp. $O(nk(n-k))$ running times are advantageous for large resp. for small values of k.

Finally, we mention in Section 4 the computation of the narrowest color-spanning strip of arbitrary orientation in time $O(n^2\alpha(k)\log k)$.

2 The smallest color-spanning circle and the farthest color Voronoi diagram

The well-known smallest enclosing circle of a set of n point sites passes through two or three of these sites. Its center is either one of the vertices of the farthest site Voronoi diagram or is the midpoint of two sites. We will define a new structure for the smallest color-spanning circle that has similar properties.

2.1 Definitions

Voronoi regions are defined as usual: Let $d(p,q)$ denote the distance from point p to point q. For a finite set, T, of sites in the plane and $a \in T$, the *Voronoi region of a in T* is defined to be $VR(a,T) := \{\ p \mid d(p,a) < \min_{b \in T \setminus \{a\}} d(p,b)\ \}$.

We have k colors, numbered from 1 to k. Let S be the set of sites and $S_1, \ldots, S_k$ the subsets for each color, i.e., $S = S_1 \dot\cup \ldots \dot\cup S_k$. Let $d_i(p) := \min_{a \in S_i} d(p,a)$ be the distance from point p to color i.

The *farthest color Voronoi region* of a site a is the set of points for which the color of a is the farthest color and a the *closest* site of this color. More formally, for a site $a \in S_j$, we say

$$FCVR(a) := \{\ p \mid d(p,a) < \min_{b \in S_j \setminus \{a\}} d(p,b) \ \text{ and } \ d_j(p) > \max_{i \neq j} d_i(p)\ \}.$$

The *farthest color Voronoi diagram FCVD* for a set of n sites with k colors is the decomposition of the plane into these *FCVR* regions. As already mentioned, it is a generalization of both the normal Voronoi diagram and the farthest site Voronoi diagram.

Lemma 1 *The smallest color-spanning circle has two or three sites, all of different colors, on its boundary. Its center is either one of the FCVD vertices or is the midpoint of two sites of different colors.*

It is clear that after computing the *FCVD*, the smallest color-spanning circle can be determined in time proportional to the complexity of the *FCVD*, i. e., the number of edges and vertices.

2.2 Structure and complexity

For each point in the plane, its distance to a site is equal to the vertical (third dimension) distance to the 45° cone with apex in that site. Hence, for each point in the plane, the minimum distance to a color is given by the lower envelope of the cones of that color. Finally, the distance to the farthest color is given by the upper envelope of the family of all the one-colored lower envelopes. The complexity of such an envelope is bounded by $O(n^{2+\varepsilon})$ and can be computed in time $O(n^{2+\varepsilon})$, see Sharir and Aggarwal [13, Theorems 7.7 and 7.16].

In the following, we will show how to improve on these bounds.

Lemma 2 *For a site $a \in S_j$ we have*

$$FCVR(a) = VR(a, S_j) \cap \overline{\bigcup_{i \neq j} VR(a, S_i \cup \{a\})}\,.$$

In other words, the *FCVR* of a is the Voronoi region of a among the sites of its own color intersected with the complement of the union of the regions of a among the sites of each other color. This follows directly from the definition, see Figure 2 for an illustration.

Lemma 3 *The complexity of a single FCVR is bounded from above by $O(n\alpha(k))$ and from below by $\Omega(n + k\alpha(k))$ which is tight for $k \in \Theta(n)$.*

Proof. Using the characterization of Lemma 2, i. e., $FCVR(a) = VR(a, S_j) \cap \overline{\cup_{i \neq j} VR(a, S_i \cap \{a\})}$, we can see that the complexity of $FCVR(a)$ is $O(n\alpha(k))$, because it is essentially the same as the complexity of the union of k convex polygons, see [13, Corollary 2.18].

For the lower bound it is clear that even a region of the farthest site Voronoi diagram, i. e. $k = n$ in our notation, can have $\Omega(n)$ vertices. Now consider an arrangement of k non-vertical line segments such that its lower envelope has complexity $\Theta(k\alpha(k))$, for the existence see [13, Theorem 4.11].

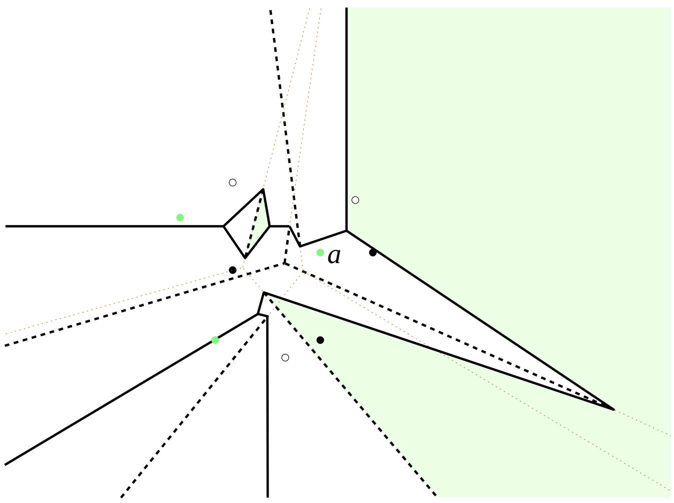

Figure 2: In this *FCVD* the thick solid resp. dashed lines mark the boundaries between two regions of different resp. the same color, while the thin dotted lines together with the dashed ones show the three normal Voronoi diagrams for the three colors. The shaded area represents the region of site a.

Place a site a sufficiently far away and above this arrangement such that each line segment is the base of a triangle which contains a in its interior and such that the lower envelope of the arrangement of the triangles has also complexity $\Theta(k\alpha(k))$, see Figure 3.

Now the first color is assigned to a. For each triangle we choose a new color, construct the three reflections of a at the three edges and assign to these three sites the new color. In this way, we get a set of $n = 3k + 1$ sites. For each color i, we know that $VR(a, S_i \cup \{a\})$ is the corresponding triangle. Then, by Lemma 2, we know that $FCVR(a)$ is the complement of the union of all triangles and has complexity $\Theta(k\alpha(k))$. □

Lemma 4 *The complexity of the FCVD is* $O(kn)$*, and for* $k \leq \frac{n}{2}$ *also* $\Omega(kn)$.

Proof. For the upper bound we count the number of connected components of all regions. This determines the complexity, since the *FCVD* is a planar graph with vertices of degree at least three to which Euler's formula applies.

As Lemma 2 has shown, the *FCVR* of a site $a \in S_j$ is the intersection of $VR(a, S_j)$ and the complement of the star-shaped polygon $\bigcup_{i \neq j} VR(a, S_i \cup \{a\})$.

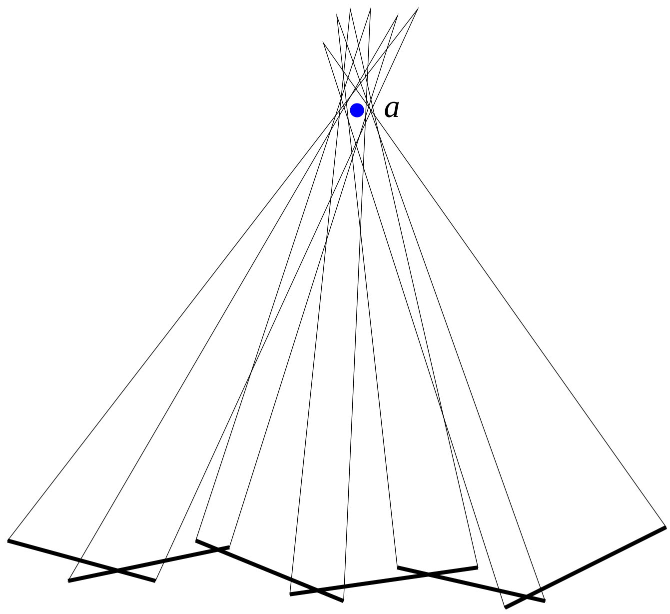

Figure 3: The construction of a lower bound for the complexity of an $FCVR$.

Some of the connected components of $FCVR(a)$ may be unbounded (only if $VR(a, S_j)$ is also unbounded, of course), and if there are more than one of them then they are separated by some also unbounded and convex $VR(a, S_i \cup \{a\})$; therefore, $FCVR(a)$ has at most $k-1$ unbounded connected components.

The bounded connected components of $FCVR(a)$ must have pieces of edges of $VR(a, S_j)$ on their back sides (as seen from a). Such an edge of $VR(a, S_j)$ intersects each (convex) $VR(a, S_i \cup \{a\})$ at most once, so it contributes at most k pieces to the $FCVD$, and in total there are $O(n)$ of them that contribute $O(kn)$ pieces which therefore also bounds the number of connected components of all regions of the $FCVD$.

The lower bound can be obtained as follows. First, for some k we construct an $FCVD$ for two sites per color with complexity $\Omega(k^2)$. This can be achieved in the following way. We place k sites on the x-axis such that the sequence of their colors is $1234\ldots\frac{k}{2}1234\ldots\frac{k}{2}$. If the sites are lifted to the paraboloid, each two tangent planes of the same color form a wedge which contributes to the $FCVD$. In the same way, another k sites are placed on the y-axis utilizing the other colors from $\frac{k}{2}+1$ to k.

The cross-like placement causes the $\frac{k}{2}$ vertical wedges to intersect the

$\frac{k}{2}$ horizontal wedges. This means that we have a pattern of $2k$ sites whose *FCVD* has complexity $\Omega(k^2)$, and the bounded regions alone already have that complexity.

Now we can arrange roughly $n/(2k)$ such patterns in the plane, far enough from each other such that the bounded regions of the patterns survive in the combined *FCVD*, which has a complexity of $n/(2k) \cdot \Omega(k^2) = \Omega(kn)$. □

2.3 Computation

Theorem 5 *The FCVD of a set of n points and k colors can be computed in $O(n^2\alpha(k)\log k)$ time.*

Proof. The construction of the diagram can be done in the following way.

1. Compute the k one-colored Voronoi diagrams in $O(\sum_{i=1}^{k} n_i \log n_i) = O(n \log n)$ time.
2. For each site a with color i do:
   (a) Locate point a in the $k-1$ different colors' one-colored Voronoi diagrams in $O(\sum_j \log n_j) = O(k \log \frac{n}{k})$ time
   (b) Construct all regions $VR(a, S_j \cup \{a\})$ in total time $O(n)$ (notice that each site cannot have more than $n-1$ neighbors, altogether).
   (c) Compute the union of the $k-1$ regions in $O(n\alpha(k)\log k)$ time [13, Corollary 6.3].
   (d) Intersect $VR(a, S_i)$ with the complement of that union. As it is starshaped and a belongs to its kernel, this can be done in time linear in the complexity of the polygons being intersected, that is, in overall $O(n\alpha(k))$ time.

Summarizing all steps, the running time of this algorithm is

$$O(n \log n + nk \log \tfrac{n}{k} + n^2\alpha(k)\log k + n^2\alpha(k)) = O(n^2\alpha(k)\log k)$$

because of $k \log \frac{n}{k} \leq n \log k$ for $k \geq 2$. □

2.4 Computing the smallest color-spanning circle without using the *FCVD*

The following result is interesting for small values of k.

Theorem 6 *Given n sites and k colors, the minimum radius circle that covers at least one site of each color can be found in $O(k^3 n \log n)$ time.*

The solution circle must have three differently colored sites in its boundary, or be determined by two diametrically opposite sites of different colors, see Lemma 1. The first case can be solved by computing the Voronoi diagram of the sites of every triple of colors, and considering the three-colored vertices of the diagrams as candidates for the center of the solution circle. Then, for each candidate, the radius of its color-spanning circle can be determined by doing point location in all the one-colored Voronoi diagrams.

The analysis of this algorithm is as follows:

1. Compute all the one-colored *VD*s in $O(\sum_{i=1}^{k} n_i \log n_i) = O(n \log n)$ time.
2. Compute all the three-colored *VD*s in $O(\sum_{i,j,m}(n_i + n_j + n_m)\log(n_i + n_j + n_m)) = O(\binom{k-1}{2}\sum_{i=1}^{k} n_i \log n) = O(k^2 n \log n)$ time.
3. Do all the point locations in $O(\sum_{i,j,m}(n_i + n_j + n_m)\sum_{p\neq i,j,m}\log n_p) = O(\sum_{i,j,m}(n_i+n_j+n_m)k \log n) = O(\binom{k-1}{2} nk \log n) = O(k^3 n \log n)$ time.

This gives an overall complexity $O(k^3 n \log n)$ for the circles defined by three points. The candidates defined by two points are computed in the analogous way, and the complexity of this part is dominated by the one above.

3 The smallest color-spanning rectangle

Now we turn to the problem for finding an axis-parallel color-spanning rectangle of minimum area or perimeter. At first sight, this problem seems to be similar to the problem of finding the smallest rectangle (or circle, convex polygon, etc.) containing at least k points from a set of n uncolored points, see e. g. Dobkin et al. [5], Aggarwal et al. [2], Eppstein and Erickson [6], or Datta et al. [4]. But their approaches cannot be extended to our setting, as we will see.

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. 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 $1 < k < n$.

3.1 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. It is clear that the smallest color-spanning rectangle, by perimeter or by area, must be non-shrinkable in the following sense.

Definition 7 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 algorithm will systematically find all candidates for non-shrinkable rectangles and compare their perimeters or areas to determine the smallest one. A starting idea to do this is shown next, this is similar to the procedure of [2].

> **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 finding the exact number of non-shrinkable rectangle, since in the worst case it seems unavoidable to check (nearly) all of them.

Lemma 8 *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 9.)

It remains to show the lower bound. To this end, we make a construction as sketched in Figure 4.

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.

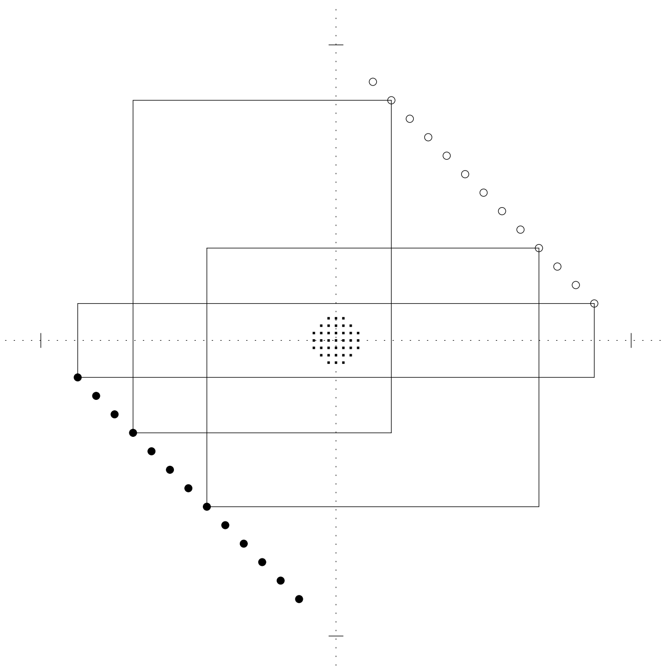

Figure 4: Each rectangle spanned by a site of color 1 and a site of color 2 is non-shrinkable.

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. □

3.2 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 horizonal 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. Note that this definition is intentionally non symmetrical in a and b, because in the next algorithm we will fix only site a while site b will be moving upwards.

Now the first condition above means that

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

The second condition can be expressed as

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

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

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

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 horizonal 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 9 *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 $\min(a_x, b_x)$ is one additional possibility. □

Now we show how to update the quantities $L_c(a, b)$ and $R_c(a, b)$ for fixed a, while b steps through the sites in y-order. Also, for each b we have to match the correct pairs of sites at the left and at the right edges. All 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) $\mathit{InclL} := \min(a_x, b_x)$;
   $\mathit{ExclL} := \max(L[a_{col}], L[b_{col}])$;
   $\mathit{InclR} := \max(a_x, b_x)$; $\mathit{ExclR} := \min(R[a_{col}], R[b_{col}])$;
   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 10 *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. □

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.

3.3 Maximal elements

Definition 11 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 $p_y < q_y$.

Note that for our special purpose we have slightly deviated from the usual definition, see [11], we are interested in maximal elements in the upper *left* (instead of right) direction, see Figure 5. 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 the y-coordinates:

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

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.

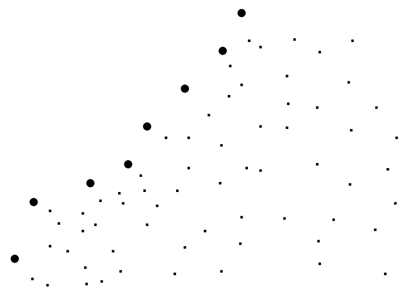

Figure 5: Maximal elements of a point set in the upper left direction.

Lemma 12 *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. □

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

Lemma 13 *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. □

And the converse is also true, in some sense.

Lemma 14 *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) from Section 3.2 hold.*

Proof. From Lemma 12 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. □

Theorem 15 *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 [11].

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 14.

So the running time of this method is $O(n(n-k)\log^2 k)$ plus the number of reported non-shrinkable rectangles but which is fortunately bounded by $O((n-k)^2)$, see Lemma 8. □

4 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. For brevity, we only state our main result.

Theorem 16 *Given n sites and k colors, the narrowest color-spanning strip can be found in $O(n^2\alpha(k)\log k)$ time.*

The proof uses techniques of inversion and outer envelopes.

5 Remarks and open problems

The location problem studied in this paper can be generalized in many ways. For example, one could ask for the smallest circle (rectangle, strip) that contains at least m (or a given percentage) of the points of each color. For circles, if we assume that every color has at least m sites, and every point p in the plane is given to the last m *significant* sites that an expanding circle centered at p would capture, the construction generalizes order-m Voronoi diagrams, which are obtained when $k = 1$. We are currently investigating this issue.

In our view, one challenging open problem is in gaining more insight into the basic farthest color Voronoi diagram introduced in this paper. For example, we would like to close the gap between the lower and upper bound for the complexity of a single region that were mentioned in Section 2.2. Also, a more efficient algorithm for constructing the *FCVD* would be highly desirable.

Finally, for the general version of the rectangle problem studied in Section 3, let us mention that the best lower bound we know is in $\Omega(n \log n)$; however, we do conjecture that the problem might be hard to solve in $o(n^2)$ time and likely belonging to the 3SUM-hard class [7] because it seems hard to avoid checking the $\Theta((n-k)^2)$ many non-shrinkable rectangles.

References

[1] P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and E. Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete Comput. Geom.*, 6(5):407–422, 1991.

[2] A. Aggarwal, H. Imai, N. Katoh, and S. Suri. Finding k points with minimum diameter and related problems. *J. Algorithms*, 12:38–56, 1991.

[3] F. Aurenhammer and R. Klein. Voronoi diagrams. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 201–290. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[4] 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.

[5] D. P. Dobkin, R. L. Drysdale, III, and L. J. Guibas. Finding smallest polygons. In F. P. Preparata, editor, *Computational Geometry*, volume 1 of *Adv. Comput. Res.*, pages 181–214. JAI Press, London, England, 1983.

[6] D. Eppstein and J. Erickson. Iterated nearest neighbors and finding minimal polytopes. *Discrete Comput. Geom.*, 11:321–350, 1994.

[7] 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.

[8] T. Graf and K. Hinrichs. Algorithms for proximity problems on colored point sets. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 420–425, 1993.

[9] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. ACM*, 31:114–127, 1984.

[10] J. S. B. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 633–701. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[11] M. H. Overmars and J. van Leeuwen. Maintenance of configurations in the plane. *J. Comput. Syst. Sci.*, 23:166–204, 1981.

[12] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.

[13] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, New York, 1995.

[14] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, volume 555 of *Lecture Notes Comput. Sci.*, pages 359–370. Springer-Verlag, 1991.