On the Rectilinear Convex Layers of a Planar Set

C. Peláez* A. Ramírez-Vigueras* C. Seara† J. Urrutia‡

Abstract

In this paper we give an optimal $O(n \log n)$ time and $O(n)$ space algorithm to compute the rectilinear convex layers of a set S of n points on the plane. We also compute the rotation of S that minimizes the number of rectilinear convex layers in $O(n^2 \log n)$ time and $O(n^2)$ space.

1 Introduction

Let $S = \{p_1, \ldots, p_n\}$ be a set of n points in the plane in general position. A *quadrant* Q of the plane is the intersection of two closed half-planes whose supporting lines are parallel to the x- and y-axes. We say that a quadrant is *S-free* if it does not contain any point of S in $int(Q)$, where $int(Q)$ denotes the interior of Q. The *rectilinear convex hull* $\mathcal{RH}(S)$ of S is defined as:

$$\mathcal{RH}(S) = \mathbb{R}^2 - \bigcup_{Q \text{ is } S\text{-free}} int(Q).$$

The rectilinear convex hull was introduced in [8] and further studied in [1] and [2]. The rectilinear convex layers of S are defined as follows:

1. The first rectilinear convex layer of S, denoted $\mathcal{L}_1$, is $\mathcal{RH}(S)$. Let S_1 denote the set of elements of S that belong to $\mathcal{RH}(S)$.

2. The i-th rectilinear convex layer $\mathcal{L}_i$ of S is the rectilinear convex hull $\mathcal{RH}(S_i)$, where $S_i = S \setminus \{S_1 \cup \cdots \cup S_{i-1}\}$, where S_j is the set of elements of S in $\mathcal{L}_j$. We stop when $S \setminus \{S_1 \cup \cdots \cup S_i\} = \emptyset$; the first i for which $S \setminus \{S_1 \cup \cdots \cup S_i\} = \emptyset$ is the number of rectilinear convex layers of S.

Figure 1 shows a point set and its rectilinear convex hull. Figure 2 shows an example of the rectilinear convex layers of a point set.

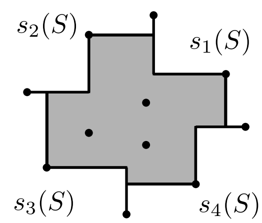


Figure 1: Rectilinear convex hull and its staircases.

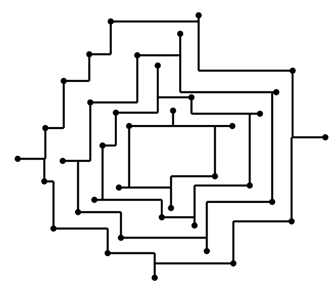
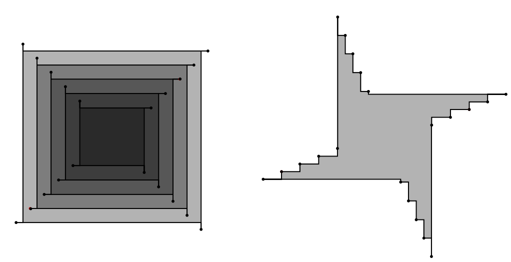


Figure 2: Rectilinear convex layers of a point set.

If we rotate S around the origin the rectilinear convex hull of S, as well as the number of rectilinear convex layers of S changes (Figure 3).



(a) A point set S with $O(n)$ rectilinear convex layers.

(b) One rectilinear convex layer of S by doing a rotation of $\alpha = \frac{\pi}{8}$.

Figure 3: As we rotate S, the number of rectilinear convex layers changes from one to $\frac{n}{4}$.

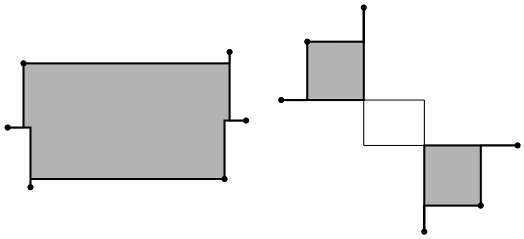
Notice that the rectilinear convex hull of S is not necessarily connected (Figure 4). It is not hard to show that if S contains at least four points in general position, there is always a rotation of S for which its rectilinear convex hull is connected, and its interior is nonempty.

Results. In this paper, we give an optimal $\Theta(n \log n)$ time and $O(n)$ space algorithm to calcu-

*Posgrado en Ciencia e Ingeniería de la Computación, Universidad Nacional Autónoma de México, `{canek, adriana.rv}@ciencias.unam.mx`. Partially supported by CONACYT of Mexico.

†Dept. de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, `carlos.seara@upc.edu`. Partially supported by projects MTM2009-07242, Gen Cat DGR2009GR1040, and the ESF EUROCORES programme EuroGIGA-ComPoSe IP04-MICINN Project EUI-EURC-2011-4306.

‡Instituto de Matemáticas, Universidad Nacional Autónoma de México, `urrutia@matem.unam.mx`. Partially supported by projects MTM2006-03909 (Spain) and SEP-CONACYT of México, Proyecto 80268.



(a) $\mathcal{RH}(S)$ with $\alpha = 0$. **(b)** $\mathcal{RH}(S)$ with $\alpha = \frac{\pi}{4}$.

Figure 4: Two different rectilinear convex hulls.

late the rectilinear convex layers of S. We also give an $O(n^2 \log n)$ time and $O(n^2)$ space algorithm to compute the rotation of S that minimizes (or maximizes) the number of rectilinear convex layers of S. We will omit some proofs due to the lack of space.

Related work. The problem of finding the convex hull of a point set S, as well as its convex layers has been studied in [9]. An optimal algorithm to find them in $O(n \log n)$ was obtained in [4].

The problem of finding the rotation of S that minimizes the area of the rectilinear convex hull was studied in [2] and in [1] it was proved that it can be found in $\Theta(n \log n)$ time and $O(n)$ space. The problem of finding the rectilinear convex hull of a point set S, is closely related to that of finding elements of S which are maximal under vector dominance. Optimal $O(n \log n)$ algorithms to find them in two and three dimensions were obtained in [6]. In [7] an $O(n^{2.688})$ algorithm for $d = n$ was given. An $O(n \log n)$ algorithm for computing the layers of maxima for point sets in three dimensions was presented in [3].

1.1 Preliminaries

For a planar point set S, we define four partial orders on S, $P(S, \preceq_i)$ using the following binary relations on pairs of elements $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$ in S:

- $p_i \preceq_1 p_j$ iff $x_i \leq x_j$ and $y_i \leq y_j$.
- $p_i \preceq_2 p_j$ iff $x_i \geq x_j$ and $y_i \leq y_j$.
- $p_i \preceq_3 p_j$ iff $x_i \geq x_j$ and $y_i \geq y_j$.
- $p_i \preceq_4 p_j$ iff $x_i \leq x_j$ and $y_i \geq y_j$.

Let $C = \{q_i = (x_i, y_i); i = 1, \ldots, r\}$ be an antichain of $P(S, \preceq_1)$ such that its elements are sorted according to their x-coordinate. C determines a *staircase* formed by the union of a set of *elbows* obtained as follows: Join q_i to q_{i+1} by an *elbow* passing trough the point (x_i, y_{i+1}), $i = 1, \ldots, r-1$ (Figure 5a). In a similar way, the antichains of $P(S, \preceq_2)$, $P(S, \preceq_3)$, and $P(S, \preceq_4)$ determine staircases (Figure 5b).

We will denote the staircase determined by the maximal elements of $P(S, \preceq_i)$ with $s_i(S)$ (Figure 1). Notice that $s_4(S)$ and $s_1(S)$, and $s_i(S)$ and $s_{i+1}(S)$ share an element of S, $i = 1, 2, 3$. The boundary of the rectilinear convex hull of S is contained in $s_1(S) \cup s_2(S) \cup s_3(S) \cup s_4(S)$, and the set of elements of S that belong to the boundary of its rectilinear convex hull, is the union of the sets of maximal elements of $P(S, \preceq_1)$, $P(S, \preceq_2)$, $P(S, \preceq_3)$, and $P(S, \preceq_4)$.

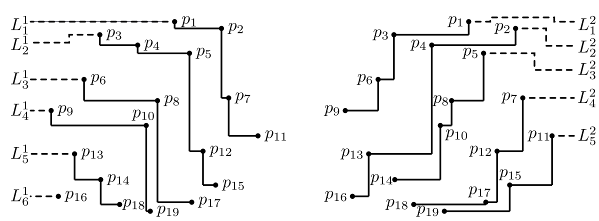
Since each of $s_i(S)$ can be obtained in $O(n \log n)$, the rectilinear convex hull of S can be obtained in the $O(n \log n)$. We will present an algorithm to compute the rectilinear convex layers of S in optimal $\Theta(n \log n)$ time. We will use the layers of maxima points as defined by Buchsbaum and Goodrich [3].

2 The k-level staircases

We define the k-level staircases of $P(S, \preceq_1)$ as follows: The 1-level staircase of $P(S, \preceq_1)$, denoted L_1^1, is $s_1(S)$. The i-level staircase of $P(S, \preceq_1)$, denoted L_i^1, is:

$$L_i^1 = s_1\left(S \setminus \bigcup_{j=1}^{i-1} L_j^1\right).$$

The k-level staircases of $P(S, \preceq_2)$, $P(S, \preceq_3)$, and $P(S, \preceq_4)$ are defined in a similar way (Figure 5).



(a) k-level staircases of $P(S, \preceq_1)$. **(b)** k-level staircases of $P(S, \preceq_2)$.

Figure 5: k-level staircases of $P(S, \preceq_1)$ and $P(S, \preceq_2)$.

We show now how to compute all the k-level staircases of $P(S, \preceq_1)$ in $O(n \log n)$ time. This problem has also been solved by Buchsbaum and Goodrich [3] with the same complexity. We present a slightly different solution using what we call a *domination tree.* This tree will allow us to solve the problem of finding the rotation of S that minimizes the number of rectilinear convex layers.

2.1 The domination tree

The *domination tree* is a rooted tree T that satisfies the following properties: The elements at depth i in T are the elements of L_i^1, ordered by the x-coordinate. The *parent* of a point $q \in L_{i+1}^1$ is a point $p \in L_i^1$, with smallest x-coordinate such that $p(x) > q(x)$ (Figure 6). To construct the domination tree, we will use an algorithm that we call the BuildTree. We present only an outline of how BuildTree works.

Assume that the elements of S are labelled $p_1, \ldots, p_n$ such that if $i < j$, then the y-coordinate of p_i is greater than the y-coordinate of p_j. We process the elements of S in this order, that is from top to bottom. Suppose that we have processed $p_1, \ldots, p_{i-1}$, and that up to this point, we have detected k layers of $S_{i-1} = \{p_1, \ldots, p_{i-1}\}$. Suppose that we have an auxiliary array A such that for any $j \leq k$, $A[j]$ contains the rightmost element of the j-th layer of S_{i-1}. The following assertion is not hard to prove: If p belongs to the r-th layer L_r^1 of S, then $A[r-1]$ dominates p_i, and $A[r]$ does not dominate p_i. Using the array A, we can find r in logarithmic time using binary search. It is not hard to see that the parent of p_i in L_{r-1}^1 can be also determined in logarithmic time.

For each point p_i, we will see if it is to the right of the rightmost point in the deepest k-level we have found: If it is so, we will add it to that k-level. Otherwise, we will find the point in the deepest level with the smallest x-coordinate greater than $p_i(x)$. If there is no such point, then p_i will go into a new k-level. Using the auxiliary sorted array A with the rightmost element of every k-level found, we will be able to do this in $O(\log n)$ time for every point, which will yield a total complexity of $O(n \log n)$ time. Thus we have the following results:

Theorem 1 *The complexity of* BuildTree *is* $O(n \log n)$.

Theorem 2 *All of the i-level staircases of $P(S, \preceq_1)$ can be constructed in $O(n \log n)$ time and $O(n)$ space.*

An example of the output of BuildTree is shown in Figure 6.

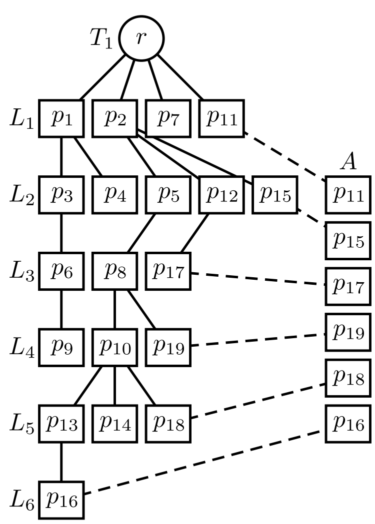


Figure 6: The tree T_1 resulting of running BuildTree on the point set shown in Figure 5.

3 Rectilinear Convex Layers

Clearly BuildTree can also be used to build all the k-level staircases of $P(S, \preceq_1)$, $P(S, \preceq_2)$, $P(S, \preceq_3)$, and $P(S, \preceq_4)$. We now show how to use them to build the rectilinear convex layers of S.

Lemma 3 *Let $S_i = S \setminus \bigcup_{k=1}^{i-1} \mathcal{L}_k$. If $p \in s_j(S_i)$ for some $j \in \{1, 2, 3, 4\}$, then p is in L_i^j.*

Proof. The result is obvious for $i = 1$. Suppose then that for some $i > 1$, $p \in \mathcal{L}_i$. Then $p \in S_i$, and thus $p \in s_j(S_i)$ for some $j \in \{1, 2, 3, 4\}$. Without loss of generality suppose that $p \in s_1(S_i)$. Then it is easy to see that there is a point $q \in s_1(S_{i-1})$ such that $p \preceq_1 q$. Therefore, p is in the i-level staircase of $P(S, \preceq_1)$. □

Corollary 4 *If $p \in \mathcal{L}_i$, then either $p \in L_i^1$, $p \in L_i^2$, $p \in L_i^3$, or $p \in L_i^4$.*

Lemma 5 *Let $p \in S$ be such that $p \in L_{j_i}^i$ for $i = 1, 2, 3, 4$. If $j = \min\{j_1, j_2, j_3, j_4\}$, then p is in the j-th rectilinear convex layer $\mathcal{L}_j$ of S.*

Theorem 6 *Let S a planar point set in general position. Computing the rectilinear convex layers of S can be done in optimal $\Theta(n \log n)$ time and $O(n)$ space.*

The time optimality follows from the fact that computing the rectilinear convex hull of a point set requires $\Omega(n \log n)$ time [1].

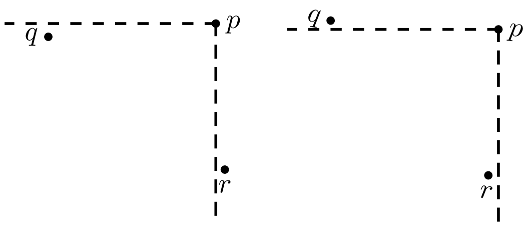
3.1 The rotation of S with the minimum number of layers

The convex hull and the convex layers of a planar point set are invariant under rotations of the point set. This is not the case when we consider the rectilinear convex hull and the rectilinear convex layers. Thus, we now study the following problem: Given a planar point set S, find a rotation of S around the origin that minimizes (or maximizes) the number of rectilinear convex layers of S.

By using the four domination trees obtained by BuildTree to compute the rectilinear convex layers of S we can produce this minimum (or maximum) in $O(n^2 \log n)$ time and $O(n^2)$ space: We only need to find the critical directions where the k-level staircases in our domination tree, and therefore the rectilinear convex layers of S, may change.

These critical directions are defined when the line joining two points in S becomes horizontal, or vertical. When this happens, the partial order $P(\preceq_i, S)$ can change, and this can cause the rectilinear convex layers and the rectilinear convex hull to change (Figure 7).

We compute first T_1, and the rectilinear convex layers of S. Next, we will generate the set $\mathcal{D}$ of critical



(a) Rotation $\alpha = 0$. **(b)** Rotation $\alpha = \frac{\pi}{24}$.

Figure 7: In the rotation $\alpha = 0$, $q \preceq_1 p$, and p and r are not comparable. In the rotation $\alpha = \frac{\pi}{24}$, q and p are not comparable, and $r \preceq_1 p$.

directions in $O(n^2)$ time and space, and sort them in $O(n^2)$ time using standard techniques. We do the sorting in such a way that when we pass through a critical direction $\gamma_i \in \mathcal{D}$ we know whether the two points p and q defining $\gamma_i \in \mathcal{D}$ become horizontal or vertical aligned according to the current orientation of the axes. Let p and q be the two points defining $\gamma_i \in \mathcal{D}$ and assume that the line joining p and q becomes horizontal (Figure 7). If p is not the parent of q or the other way around, we will continue to the next critical direction. Otherwise, q jumps from the j-level staircase to the $(j-1)$-level staircase, so we remove q from its staircase it belongs to, and add it to that one level above it. We repeat this recursively for all the descendants of q.

If the line joining p to q becomes vertical, we proceed as follows: If both p and q are not in the same j-level staircase and next to each other in that staircase, continue to the next critical direction. Otherwise, suppose that p is before r in their j-level staircase. Then r jumps down from the j-level staircase to the $(j+1)$-level staircase: We create the $(j+1)$-level staircase if it does not exists.

Lemma 7 *When going from γ_i to γ_{i+1}, a point $p \in S$ either preserves its depth, or it only changes by* 1.

Lemma 8 *When a point q jumps up from the j-level staircase to the $(j-1)$-level staircase, all its descendants in the T_1 tree also jump up one staircase level, and not other point in S does.*

The result is similar for the other three trees.

Theorem 9 *At the end of the i-th cycle of the loop, the L_j array will be the j-level staircase L_j^1 of S in the rotation γ_i.*

To prove the time complexity of the algorithm we need the following non trivial lemma, given without proof.

Lemma 10 *The number of times $p \in S$ changes its rectilinear depth as S rotates is at most $O(n)$.*

Theorem 11 *Let S a planar point set in general position. Computing the rotation of S that minimizes the number of layers can be done in $O(n^2 \log n)$ time and $O(n^2)$ space.*

This algorithm can be modified to solve other problems such as computing the rotation that: (1) maximizes the number of rectilinear convex layers of S, (2) produces the rectilinear convex hull with the largest (smallest) number of vertices.

References

[1] C. Alegría-Galicia, T. Garduño, A. Rosas-Navarrete, C. Seara, and J. Urrutia. Rectilinear convex hull with minimum area. *The XIV Spanish Meeting on Computational Geometry*, 2011.

[2] S. W. Bae, C. Lee, H. K. Ahn, S. Choi, K. Y. Chwa. Computing minimum-area rectilinear convex hull and L-shape. *Computational Geometry: Theory and Applications*, 42(9), 903–912, 2009.

[3] A. L. Buchsbaum, and M. T. Goodrich. Three-dimensional layers of maxima. *Algorithmica*, 39(4), 275–286, 2004.

[4] B. Chazelle. On the convex layers of a planar set. *IEEE Transactions on Information Theory*, 31, 509–517, 1985.

[5] B. Chazelle, L. Guibas, and D. T. Lee. The power of geometric duality. *Proc 24th IEEE Ann. Symp. Foundations of Computer Science*, 217–225, 1983.

[6] H. T. Kung, F. Luccio, and F. P. Preparata. On finding the maxima of a set of vectors. *J. ACM*, 22(4), 469–476, 1975.

[7] J. Matoušek. Computing dominances in E^n. *Information Processing Letters*, 38(5), 277–278, 1991.

[8] T. Ottmann, E. Soisalon-Soininen, and D. Wood. On the definition and computation of rectilinear convex hulls. *Information Sciences*, 33(3), 157–171, 1984.

[9] M. H. Overmars, and J. van Leeuwen. Maintenance of configurations in the plane. *Journal of Computer and System Sciences*, 23(2), 166–204, 1981.

[10] C. Peláez, A. Ramírez-Vigueras, C. Seara, and J. Urrutia. Weak separators, vector dominance, and the dual space. *XIV Spanish Meeting on Computational Geometry*, 2011.

[11] M. I. Shamos. Computational Geometry. *Ph. D. dissertation, Yale Univ., New Haven, CT, 1978.*