Separating bichromatic point sets by L-shapes

Farnaz Sheikhia, Ali Mohades, Mark de Berg, Mansoor Davoodi

Article history:
Received 21 December 2012
Accepted 3 February 2014
Available online 29 June 2015

Keywords:
Separability
Bichromatic point sets
L-shape

ABSTRACT

Given a set R of red points and a set B of blue points in the plane, we study the problem of determining all angles for which there exists an L-shape containing all points from B and no points from R. We propose a worst-case optimal algorithm to solve this problem in O(n^2) time and O(n) storage, where n = |R| + |B|. We also describe an output-sensitive algorithm that reports these angles in O(n^{8/5+\varepsilon} + k \log k) time and O(n^{8/5+\varepsilon}) storage, where k is the number of reported angular intervals and \varepsilon > 0 is any fixed constant.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Background. In a separability problem in \(\mathbb{R}^2\) one is given two colored sets of objects, a set R of red objects and a set B of blue objects, and a class \(\mathcal{S}\) of curves. The curves are usually either infinite curves (such as lines) or closed curves (such as circles), so that they partition the plane into two regions. The goal is now to decide whether there is a curve in \(\mathcal{S}\) that is a separator for R and B, that is, a curve that partitions the plane such that R and B lie in different regions. If such a placement is possible, one often also wants to compute all separators, or the separator minimizing some cost function. Geometric separability arises in applications where classification is required, such as machine learning and image processing.

There has been a fair amount of work on different kinds of separators, both in the plane and in higher dimensions. For separability in the plane where the objects to be separated are points—this is the topic of our paper—the following results are known. The most basic version of the problem is where the class \(\mathcal{S}\) is the class of all lines. The problem of deciding whether the two point sets can be separated by a line can be solved in linear time, as shown by Megiddo [14]. Seara [18] showed how to compute in linear time all orientations for which there exists a line separating the two point sets. O’Rourke et al. [17] studied a different type of separators, namely circles. They presented a linear-time algorithm for deciding whether the two point sets can be separated by a circle, and they also gave algorithms for finding the smallest and the largest separating circle. The problem of finding a convex polygon with minimum number of edges separating the two point sets, if it exists, was solved by Edelsbrunner and Preparata [6] in \(O(n \log n)\) time. Fekete [7] showed that the problem of determining a simple polygon with a minimum number of edges separating the two point sets is NP-hard, and a polynomial-time approximation algorithm was provided by Mitchell [15]. Separability problems have been studied for separators in the form of strips and wedges [9] as well. A thorough study is presented by Seara [18].

* Corresponding author.
E-mail addresses: f.sheikhia@aut.ac.ir (F. Sheikhia), mohades@aut.ac.ir (A. Mohades), mbberg@win.tue.nl (M. de Berg), mdmonfared@iasbs.ac.ir (M. Davoodi).

http://dx.doi.org/10.1016/j.comgeo.2015.06.008
0925-7721/© 2015 Elsevier B.V. All rights reserved.
Separability has also been studied for rectangular separators. This has applications in urban scene reconstruction, which seeks to reconstruct buildings from LiDAR data. The idea is, roughly, to first cluster the data points, then project the points from each cluster onto a suitable plane, and finally find the building facets (typically walls and roofs) corresponding to the clusters [11–13]. Since walls and roofs are often rectangles or other rectilinear shapes, the task is to find a suitable rectilinear shape enclosing the points. Sometimes there are also points that are known to not be part of the facet being reconstructed. We then seek a shape that includes the points in the facet (positive samples) while excluding the points known to not be in the facet (negative samples). The distinction between the positive and negative samples is represented by the color assigned to them, thus leading to separability problems for rectilinear shapes. Following this motivation, Van Kreveld et al. [12,13] recently studied the separability problem in the plane for the case where the separator is a (not necessarily axis-aligned) rectangle. They proposed an $O(n \log n)$ time algorithm to compute all orientations for which a rectangular separator exists. They also required that the rectangle tightly fits the blue point set. More precisely, the rectangle should be $\delta$-covered by the blue points, meaning that it is contained in the union of the radius-$\delta$ disks centered at the blue points. They mentioned the case of a non-convex separator, namely an L-shape, as an open problem. This is the topic of our paper, except that we do not require the L-shape to be $\delta$-covered. Extending it to take this extra condition into account is an interesting open problem. Next we define the problem more precisely and we state our results.

Exact problem statement and results. We define an axis-aligned L-shape to be the set-theoretic difference $M \setminus M'$ of two axis-aligned rectangles $M$ and $M'$ such that the top-right corners of $M$ and $M'$ coincide. More precisely, an L-shape is defined as $\text{Cl} (\text{Int}(M) \setminus \text{Int}(M'))$, where $M$ and $M'$ are closed rectangles with $M' \subseteq M$ that share their top-right corner. (Here $\text{Cl}(\cdot)$ and $\text{Int}(\cdot)$ denote the closure and interior, respectively, of a planar set.) Note that an L-shape is a closed set, that is, it includes its boundary and that an L-shape can degenerate into a rectangle. An L-shape with orientation $\theta$ is defined as an axis-aligned L-shape that has been rotated in counterclockwise direction over an angle of $\theta$. Now, given a blue point set $B$ and a red point set $R$, we wish to find all angles $\theta \in [0, 2\pi)$ for which there exists an L-shape $L$ with orientation $\theta$ such that $B \subseteq L$ and $R \cap L = \emptyset$. From now on, we call such an L-shape a blue L-shape. The orientations for which a blue L-shape exists form a collection of subintervals of $[0, 2\pi]$. We present two algorithms for computing this collection of intervals.

The first algorithm is an algorithm running in $O(n^2)$ time and using $O(n)$ storage. We prove that this algorithm is worst-case optimal by showing that there are point sets that admit $\Omega(n^2)$ disjoint intervals for which there exists a blue L-shape. The second algorithm is an output-sensitive algorithm which uses $O(n^{\frac{5}{2} + \varepsilon} + k \log k)$ time and $O(n^{\frac{5}{2} + \varepsilon})$ storage, where $k$ is the number of reported angular intervals and $\varepsilon > 0$ is any fixed constant. (Our results are based on two preliminary papers [20,21]. Compared to these preliminary results, the running time of the first algorithm has been slightly improved so that it is now worst-case optimal. The lower bound is completely new.)

2. Preliminaries

**Terminology and notation.** Our global strategy will be to do a rotational sweep: we increase $\theta$ from 0 to $2\pi$ and we report the angular intervals for which there is a blue L-shape while we sweep. It will be convenient to think about the sweep as rotating the coordinate frame. Thus, we define the $x_0$-axis and the $y_0$-axis as the coordinate axes after the coordinate frame has been rotated over an angle $\theta$ in counterclockwise direction. We denote the coordinates of a point $p$ in the rotated coordinate frame by $x_0(p)$ and $y_0(p)$. Whenever we talk about the top-right corner of a rectangle, we mean the top-right corner with respect to the current coordinate frame.

For an angle $\theta$, we denote the minimum bounding rectangle with orientation $\theta$ of the blue point set $B$ by $M_B(\theta)$. In other words, $M_B(\theta)$ is the axis-aligned bounding box of $B$ in the rotated coordinate frame. Let $R_\theta := R \cap M_B(\theta)$ be the set of red points inside $M_B(\theta)$, and define $M_{B}(\theta)$ to be the smallest rectangle with orientation $\theta$ that contains $R_\theta$ and shares its top-right corner with $M_B(\theta)$. Note that $L_\theta := M_B(\theta) \setminus M_{B}(\theta)$ is an L-shape.

To determine whether $L_\theta$ contains all points from $B$, we will define a so-called step-shape. We say that a point $q$ dominates another point $p$ (at orientation $\theta$) if $x_0(q) \geq x_0(p)$ and $y_0(q) \geq y_0(p)$, and we say that a point $b_1 \in B$ is maximal (at orientation $\theta$) if there is no other point $b_2 \in B$ that dominates $b_1$. The step-shape is now defined as the region in $M_B(\theta)$ consisting of all points that are dominated by a blue point. The step-shape is a closed shape, bounded by the staircase that connects the maximal blue points, and by parts of the boundary of $M_B(\theta)$—see Fig. 1. Using the step-shape, we can characterize when $L_\theta$ contains all the blue points. To this end, we define the red witness as the lower-left corner of $M_B(\theta)$; note that the red witness is the reflex corner of $L_\theta$. We denote the red witness by $w_\theta$. Then $L_\theta$ contains all blue points if and only if the red witness $w_\theta$ lies outside or on the step-shape. However, note that despite containing all blue points in this case, $L_\theta$ is not blue as it contains red points on its boundary. But, here we can slightly shrink $L_\theta$ (by slightly enlarging $M_B(\theta)$) to obtain an L-shape not containing any red points. The shrunk L-shape still contains all blue points if and only if the red witness did not lie on the boundary of the step-shape. So we have the following observation.

**Observation 1.** There exists a blue L-shape in the orientation $\theta$ if and only if the red witness $w_\theta$ lies outside the step-shape.

**The global strategy.** As remarked earlier, our global strategy is to perform a rotational sweep. While we sweep, we will maintain the following information:
• The rectangle $M_B(\theta)$. More precisely, we maintain the set of blue points on the boundary of $M_B(\theta)$. (Except at events there are at most four such points.)
• The step-shape or, more precisely, the staircase bounding the step-shape from the right.
• The red witness $w_\theta$, or, more precisely, the two red points defining $w_\theta$.

While we do the sweep, we are interested in the angles where the red witness crosses the staircase; these angles define the angular intervals we have to report. Next we discuss the various events that arise during the sweep.

**Blue-rect event:** this event occurs when the set of blue points defining $M_B(\theta)$ changes. This happens when the blue point with the maximum or minimum $x_0$-coordinate (or the maximum or minimum $y_\theta$-coordinate) changes.

**Blue-sc event:** this event occurs when the staircase changes, that is, when the set of maximal blue points changes.

**Red-set event:** this event occurs when $R_\theta$, the set of red points inside $M_B(\theta)$, changes.

**Witness event:** this event occurs when the points defining the red witness $w_\theta$ change. This happens when the red point in $R_\theta$ with the minimum $x_0$-coordinate changes, and when the red point in $R_\theta$ with the minimum $y_\theta$-coordinate changes. It can be triggered by a red-set event, or by two red points that were already in $R_\theta$ swapping order along the $x_0$-axis or $y_\theta$-axis.

**Crossing event:** this event occurs when the red witness $w_\theta$ crosses the staircase.

Note that all these events take place at angles defined by pairs of points: blue-rect events and blue-sc events take place at angles defined by two blue points, red-set events take place at angles defined by a red and a blue point, witness events occur at angles defined by a red and a blue or two red points, and crossing events occur at angles defined by a red and a blue point. Thus, one way to compute all angles for which there exists a blue L-shape is to consider all $\Theta(n^2)$ angles defined by a pair of points in $B \cup R$ (even though not all such angles necessarily define an event), sorting them to get $\Theta(n^2)$ subintervals of $[0, 2\pi)$ in order, and examining whether there exists a blue L-shape in each of these subintervals. In preliminary versions [20,21] of this paper it has been shown that this approach leads to algorithms that use $O(n)$ storage and have $O(n^2 \log n)$ [20] or $O(n^2 \alpha(n))$ [21] running time, where $\alpha(n)$ is the extremely slow-growing inverse of the Ackermann function [19]. In the present paper we will give an algorithm with running time $O(n^2)$, while still using only $O(n)$ storage.

### 3. A worst-case optimal algorithm

The idea of the first algorithm is to pre-compute all blue-rect, blue-sc, red-set, and witness events (from now on, we refer to these events as non-crossing events), and then compute all the crossing events occurring between two consecutive non-crossing events. In the following subsections, we first describe how to pre-compute the non-crossing events, then we describe an algorithm that achieves the optimal $O(n^2)$ time complexity but uses $O(n \alpha(n))$ storage, and finally we refine the algorithm so that it uses only linear storage.

#### 3.1. Pre-computing non-crossing events

Computing the blue-rect events is easy: after computing the convex hull of the blue points in $O(n \log n)$ time [4], we can find the blue-rect events by using rotating calipers [22], leading to the following lemma.

**Lemma 2.** There are $O(n)$ blue-rect events in total, and they can be computed in $O(n \log n)$ time and using $O(n)$ storage.

Next we turn our attention to the blue-sc events. Recall that these events occur when the set of maximal blue points changes, as we rotate the coordinate frame. Bae et al. [3] have proved that the set of maximal points changes at most $O(n)$ times during a full rotation from $\theta = 0$ to $\theta = 2\pi$; they also presented an algorithm to compute all these changes in $O(n^2)$ time, using $O(n)$ storage. However, we can also use an algorithm by Avis et al. [2], who studied the same problem; their algorithm runs in $O(n \log n)$ time and $O(n)$ storage. We can conclude the following lemma.
Lemma 3. There are $O(n)$ blue-sc events in total, and they can be computed in $O(n \log n)$ time and using $O(n)$ storage.

To find the red-set events—these are the events where a red point $r$ enters or exits the rectangle $M_B(\theta)$—we observe that they occur at angles defined by the tangent lines from $r$ to the convex hull of the blue points [12]. We denote this convex hull by $CH_B$. (If $r$ lies inside $CH_B$, then it will always be in $R_\theta$ and consequently in the rectangle $M_B(\theta)$. Hence, these points can be ignored.) After computing $CH_B$, we can test if $r$ lies inside $CH_B$ in $O(\log n)$ time and, if not, we can compute the tangent lines from $r$ in $O(\log n)$ time. A red point $r$ which is outside $CH_B$ can enter or exit $M_B(\theta)$ if the angle between the two tangents from $r$ to $CH_B$ is greater than or equal to $\pi/2$. Thus, we only consider these red points, ignoring the rest. Let $r$ be such a red point, and as illustrated in Fig. 2, let $\ell$ be a tangent from $r$ to $CH_B$. Now define $\theta$ to be the counterclockwise angle that $\ell$ makes with the positive $x$-axis. Then, the angles $(\theta + i \cdot (\pi/2))$ mod $2\pi$, for $i = 0, 1, 2, 3$, correspond to red-set events. Let $\ell'$ be the other tangent from $r$ to $CH_B$. Then, the red-set events corresponding to $\ell'$ can be computed similarly. The only difference is that in this case we take $\theta$ as the counterclockwise angle that $\ell'$ makes with the positive $y$-axis. This leads to the following lemma.

Lemma 4. There are $O(n)$ red-set events in total, and they can be computed in $O(n \log n)$ time and using $O(n)$ storage.

To compute the witness events, we proceed as follows. The red witness $w_\theta$ is defined by the red point in $R_\theta$ with the minimum $x_\theta$-coordinate and the red point in $R_\theta$ with the minimum $y_\theta$-coordinate. For each red point $r$, define the function $f_r(\theta)$ as follows:

$$f_r(\theta) = \begin{cases} x_\theta(r) & \text{if } r \in R_\theta \\ \text{undefined} & \text{otherwise} \end{cases}$$

Recall that the angles at which a red point $r$ may enter or leave the rectangle $M_B(\theta)$, and thus become or cease to be a point of $R_\theta$, correspond to the tangents from $r$ to $CH_B$. Each of these two tangent lines can induce four events where $r$ enters or leaves $M_B(\theta)$. (This is the case because $M_B(\theta) = M_B(\theta + \pi/2) = M_B(\theta + \pi) = M_B(\theta + 3\pi/2)$. We still have to do a full rotation, however, because the top-right corner of $M_B(\theta)$ is different in each of these rotated coordinate frames.) So, we can have eight events in total, partitioning $[0, 2\pi)$ into nine intervals, at most five of which can correspond to angles where $r$ is inside $M_B(\theta)$. Hence, $f_r(\theta)$ consists of at most five pieces, and after having computed all red-set events we can compute each $f_r(\theta)$ in $O(1)$ time. So, over all points $r \in R$ we have $O(n)$ pieces of functions. The point of $R_\theta$ with the minimum $x_\theta$-coordinate is given by the lower envelope of these functions. Note that any two pieces intersect in at most one point. Namely, we can only have $f_{r_1}(\theta) = f_{r_2}(\theta)$ if $\theta$ is the orientation orthogonal$^1$ to the line through $r$ and $r'$.

It is known that the complexity of the lower envelope of $O(n)$ curves such that any two curves intersect in at most one point is $O(n\alpha(n))$ [19]. Hence, the minimum $x_\theta$-coordinate changes $O(n\alpha(n))$ times. We can find these events by computing the lower envelope in $O(n \log n)$ time [8]. Similarly, the minimum $y_\theta$-coordinate changes $O(n\alpha(n))$ times, and we can find the changes in $O(n \log n)$ time. We get the following lemma.

Lemma 5. There are $O(n\alpha(n))$ witness events in total, and they can be computed in $O(n \log n)$ time and using $O(n\alpha(n))$ storage.

3.2. A quadratic-time algorithm using $O(n\alpha(n))$ storage

Our algorithm works roughly as follows. After pre-computing all non-crossing events using Lemmas 2–5, we sort them in increasing order of the angle at which they occur. To initialize the rotational sweep, we set $\theta := 0$ and we initialize $M_B(\theta)$, the staircase, and the red witness $w_\theta$. The most time consuming part of the initialization is the computation of the

$^1$ In fact, there are two opposite orientations orthogonal to this line. Thus, the pieces can intersect twice. However, by treating the angles $[0, \pi)$ and $(\pi, 2\pi)$ separately, we can reduce this to the case of a single intersection.
maximal blue points—the points defining the staircase—which takes $O(n \log n)$ time. Now we start our rotational sweep. We go through the non-crossing events in order, and handle them according to their type: if the event is a blue-rect event we update $M_B(\theta)$, if it is a blue-sc event we update the staircase, and if it is a witness event then we update the red witness. Our task is now to detect the crossing events occurring between two consecutive non-crossing events. Recall that the crossing events occur when the red witness crosses the horizontal or the vertical (with respect to the current coordinate frame) edges of the staircase. Thus, potential crossing events take place when the red witness crosses the horizontal or the vertical (with respect to the current coordinate frame) lines through the points defining the staircase. In other words, potential crossing events take place at angles defined by a red point and a blue point, where the red point is one of the points defining the red witness and the blue point is one of the maximal blue points in the current coordinate frame. We have pre-computed the non-crossing events, but we cannot afford to pre-compute and sort all $\Theta(n^2)$ potential crossing events. To solve this we proceed as follows. Let $\theta_1, \theta_2, \ldots, \theta_k$ be the sorted sequence of angles corresponding to the blue-rect events, the blue-sc events, and the red-set events—we do not use the witness events here, except for the ones that are also red-set events—and define $\theta_0 := 0$ and $\theta_{k+1} = 2\pi$. The angles $\theta_i$ partition $[0, 2\pi)$ into $O(n)$ sweep intervals, and we will detect the crossing events in each sweep interval using a grid-based approach.

**Computing crossing events in a sweep interval.** Let $[\theta_i, \theta_{i+1})$ be the current sweep interval to process. The crossing events in $[\theta_i, \theta_{i+1})$ are detected as follows.

First we process the event corresponding to $\theta_i$: if $\theta_i$ corresponds to a blue-rect event then we update $M_B(\theta)$, if it corresponds to a blue-sc event then we update the staircase, and if it corresponds to a red-set event that is also a witness event then we update the red witness $w_\theta$.

Now we let $\theta$ increase from $\theta_i$ to $\theta_{i+1}$. As $\theta$ increases, the witness point $w_\theta$ and the staircase move, and we need to detect the events where $w_\theta$ crosses the staircase. Consider the grid $G_\theta$ inside $M_B(\theta)$ that is defined by the lines through the edges of the blue staircase, as shown in Fig. 3.

As $\theta$ increases, the grid lines move. Since there are no blue-sc events inside the sweep interval $[\theta_i, \theta_{i+1})$, the grid lines move in a continuous fashion. Similarly, $w_\theta$ moves in a continuous fashion: there can be witness events inside $[\theta_i, \theta_{i+1})$, but no witness events triggered by red-set events, and this implies that $w_\theta$ moves continuously. Because the grid $G_\theta$ and the witness $w_\theta$ move continuously as $\theta$ increases from $\theta_i$ to $\theta_{i+1}$, we can easily trace $w_\theta$ through the grid and detect the crossing events. The details are explained next.

Define $C_\theta$ as the cell of $G_\theta$ containing $w_\theta$. First we determine $C_\theta$, that is, the grid cell that contains $w_\theta$ at the start of the sweep interval $[\theta_i, \theta_{i+1})$. Let $b_1, b_2, \ldots$ be the maximal blue points during $[\theta_i, \theta_{i+1})$, numbered from left to right. Determining $C_\theta$ can be done in $O(\log n)$ time by two binary searches on the staircase, one using the $x_0$-coordinate and the other using the $y_0$-coordinate of $w_\theta$. Tracing $w_\theta$ through the grid means keeping track of the cell $C_\theta$. Note that all crossing events take place at angles where $C_\theta$ changes, and that it is easy to check whether such a change corresponds to a crossing event.

Now suppose that we know the current cell $C_\theta$ and we wish to know when $C_\theta$ changes, that is, when $w_\theta$ moves to a neighboring cell. Let $b_j$ and $b_{j+1}$ be the maximal blue points determining the top and bottom of $C_\theta$, let $b_j^r$ and $b_{j+1}^r$ be the maximal blue points determining the left and right side of $C_\theta$, and let $r$ and $r'$ be the red points defining $w_\theta$—see Fig. 3. Using the points $b_j, b_{j+1}, b_j^r, b_{j+1}^r$ and $r, r'$, we can determine in $O(1)$ time the angle $\theta^*$ at which $w_\theta$ leaves $C_\theta$. Indeed, the angle $\theta^*$ is the minimum among the four potential angles: the angle defined by the line $\ell(b_j, r)$, the angle defined by the line $\ell(b_{j+1}, r)$, the angle defined by the line orthogonal to $\ell(b_j^r, r')$, and the angle defined by the line orthogonal to $\ell(b_{j+1}^r, r')$. (Here $\ell(p, q)$ denotes the line through two given points $p, q$.)

Recall, however, that the points $r, r'$ defining $w_\theta$ can change within a sweep interval. When this happens before the angle $\theta^*$ is reached, then the computation above is invalid. Thus, apart from $\theta^*$, we also need to consider $\theta_{\text{witness}}$, which is the angle corresponding to the next witness event. In addition, we need to take $\theta_{i+1}$, the angle where the current sweep interval ends, into account. Recall that we have pre-computed the witness events and other non-crossing events, and kept...
them in order. Thus, using these sorted lists we can have access to $\theta_{\text{witness}}$ in $O(1)$ time. Having $\theta^*, \theta_{\text{witness}}$, and $\theta_{i+1}$ available, the next event to handle is the one which occurs soonest. Three cases arise:

**Case 1:** the angle $\theta^*$ is the angle corresponding to the next event. In this case, $w_\theta$ crosses the boundary of $C_\theta$ at the angle $\theta^*$ before any other event. The angle $\theta^*$ corresponds to a potential crossing event. To handle this event first we update $C_\theta$: if $w_\theta$ crosses the line defined by $b_j$ ($b_{j+1}$, or $b_{j'}$, or $b_{j'+1}$), then we set $j := j - 1$ (or $j := j + 1$, or $j' := j' - 1$, or $j' := j' + 1$, respectively). This takes $O(1)$ time. We can also test in $O(1)$ time whether we have a crossing event. If so, a new output interval gets started when $w_\theta$ crosses the staircase from below to above, and the current output interval gets terminated when $w_\theta$ crosses the staircase from above to below.

Finally, we repeat the procedure in the new cell $C_\theta$. (This involves computing the new angle $\theta^*$ at which $w_\theta$ leaves the new cell $C_\theta$, that is, repeating the computation with the updated values $j, j'$.)

**Case 2:** the angle $\theta_{\text{witness}}$ is the angle corresponding to the next event. In this case, we update the points $r, r'$ defining $w_\theta$, and we recompute the angle $\theta^*$ using the new points $r, r'$. This takes $O(1)$ time. We also need to go to the next witness event in the list of the pre-computed witness events to update $\theta_{\text{witness}}$. This takes $O(1)$ time as well. We continue the procedure in the same cell $C_\theta$ by handling the next event which will be determined by comparing the angle $\theta_{i+1}$ and the new angles $\theta^*$ and $\theta_{\text{witness}}$.

**Case 3:** the angle $\theta_{i+1}$ is the angle corresponding to the next event. In this case, $w_\theta$ stays in the same cell until the end of $[\theta_i, \theta_{i+1})$. Thus, we are done with the current sweep interval, and can proceed to process $[\theta_{i+1}, \theta_{i+2})$. Handling this case takes $O(1)$ time as well.

To summarize, by tracing $w_\theta$ through the grid $G_\theta$, we can find the crossing events in the sweep interval $[\theta_i, \theta_{i+1})$ in order, and output the subintervals of $[\theta_i, \theta_{i+1})$ for which there exists a blue L-shape. After having done this for all of the $O(n)$ sweep intervals we are done.

Note that the output intervals that are reported are all open since the red witness is not allowed to be on the staircase. Recall that we report the output intervals sweep interval by sweep interval. Now, besides the output intervals that have both endpoints in a single sweep interval, there may be output intervals that start in one sweep interval but end in another sweep interval. Thus, when we reach the end of a sweep interval, there can be an output interval that is still pending. Such an interval will be completed when its endpoint (a crossing event) is reached in a later sweep interval, and this output interval will be reported at that time.

**Degenerate cases.** Before we analyze the running time of the algorithm, we briefly discuss the case when there are three or more collinear points, and several events happen simultaneously. This is, in fact, easy to deal with: We use the “simulation of simplicity (SoS)” technique [5], so we can proceed as if we do not have degenerate cases. (Intuitively, it means that ties are broken arbitrarily, but in a globally consistent manner.) Then we get a number of output intervals in each sweep interval. As explained earlier these output intervals are all open. The ones that have “infinitesimal length” are caused by SoS. Thus, by a post-processing in each sweep interval we discard the output intervals with “infinitesimal length”, and report the rest. This way we can easily deal with degeneracies.

**Complexity analysis.** Pre-computing all non-crossing events (blue-rect, blue-sc, red-set, and witness events) takes $O(n \log n)$ time by Lemmas 2–5, and sorting them can be done in the same amount of time. After this, the rotational sweep that detects the crossing events starts. The sweep is done in $O(n)$ sweep intervals, defined by the blue-rect, blue-sc, and red-set events. The initialization at the start of each sweep interval takes $O(\log n)$ time, to determine the grid cell containing the red witness $w_\theta$. Each event within a sweep interval takes $O(1)$ time. The total number of events over all sweep intervals is $O(n^2)$, because each event corresponds to the angle defined by a pair of points. Hence, the total time for the whole algorithm is $O(n^2 \log n)$. The total amount of working storage is dominated by the storage we need to store the $O(n^2 \log n)$ pre-computed witness events. This leads to the following result.

**Lemma 6.** Let $B$ be a set of blue points and let $R$ be a set of red points in the plane, with $n := |B| + |R|$. Then, we can compute all angular intervals for which there is a blue L-shape in $O(n^2)$ time and using $O(n^2 \log n)$ storage.

### 3.3. A quadratic-time algorithm using linear storage

The algorithm described so far uses $O(n \log n)$ storage because it pre-computes and stores all witness events. We need the witness events because within a sweep interval we want to trace the red witness $w_\theta$ through the grid $G_\theta$. Instead of pre-computing the witness events, we can also proceed as follows.

Recall that the red witness is defined by $r, r'$, which are the points of $R_\theta$ with minimum $y_{\theta^*}$ and $x_{\theta^*}$-coordinate, respectively. The witness events are the events where $r$ or $r'$ changes. Obviously, $r$ and $r'$ are vertices of $\mathcal{CH}(R_\theta)$, the convex hull of $R_\theta$. Now we observe that $R_\theta$ does not change within a sweep interval, because the red-set events are used (together with the blue-rect and blue-sc events) to define the sweep intervals. This implies we can simply proceed as follows. Consider the point $r$; the point $r'$ can be dealt with in a similar manner. At the start of the sweep interval $[\theta_i, \theta_{i+1})$ we find the vertex of
4. The lower bound

In the previous section we described an $O(n^2)$ time algorithm to find, given a blue point set $B$ and a red point set $R$ of total size $n$, all subintervals of $[0, 2\pi)$ for which there exists a blue L-shape. We call such intervals blue subintervals from now on. In this section we show that our algorithm is worst-case optimal by describing sets $B$ and $R$ for which the number of blue intervals is $\Omega(n^2)$. To slightly simplify notation, the total number of points used in the construction will be $3n + 5$, rather than just $n$; clearly this is sufficient to prove the asymptotic lower bound.

The blue point set $B = \{b_0, \ldots, b_{n+4}\}$ is defined as follows; see also Fig. 4.

- For $0 \leq i \leq n$, we have $b_i = (i, -i)$.
- Furthermore, $b_{n+1} = (n^4 + 2n, -n^4 - 2n)$, $b_{n+2} = (-n^4 - n, -3n^4 - 5n)$, $b_{n+3} = (-3n^4 - 4n, -n^4 - 2n)$, and $b_{n+4} = (-n^4 - n, n^4 + n)$.

**Observation 8.** For $0 \leq \theta < \pi/4$, the bounding rectangle $M_B(\theta)$ is defined by the points $b_{n+1}, \ldots, b_{n+4}$, and the blue staircase consists of the points $b_{n+4}, b_0, \ldots, b_n, b_{n+1}$. Thus, there are no blue-rect events and no blue-sc events during the angular interval $[0, \pi/4)$.

The idea of the overall construction is to place $n$ pairs of red points such that each pair induces $\Omega(n)$ blue intervals as $\theta$ increases from $0$ to $\pi/4$. More precisely, there will be pairs $r_j, r'_j$ for $j = 1, \ldots, n$, and angles $0 = \theta_0 < \theta_1 < \ldots < \theta_n < \pi/4$ such that
Theorem 6. Suppose that $B_j, D_j > 0$, and that

\[
\frac{i \varepsilon_j}{B_j - i \varepsilon_j} < \frac{1 + i \varepsilon_j'}{D_j + i \varepsilon_j} < \frac{(i + 1) \varepsilon_j}{B_j - (i + 1) \varepsilon_j'} \quad \text{for all } i = 0, 1, \ldots, n - 1,
\]

and that

\[
\arctan \left( \frac{ne_j}{B_j - ne_j'} \right) \leq \theta_j - \theta_{j-1}.
\]

Then the red witness $w_j := (x_0(r_j'), y_0(r_j))$ crosses all the vertical and horizontal staircase edges in between $b_0$ and $b_n$ in order from top to bottom as $\theta$ increases from $\theta_{j-1}$ to $\theta_j$.

Proof. Let $s_v(b_i, \theta)$ and $s_h(b_i, \theta)$ be the vertical and horizontal staircase steps incident to $b_i$ at angle $\theta$ respectively, and let $\ell_v(b_i, \theta)$ and $\ell_h(b_i, \theta)$ be the line containing these steps respectively. The line $\ell_h(b_i, \theta)$ contains the red point $r_j$ at angle $\theta = \theta_{j-1} + \arctan((i \varepsilon_j)/ (B_j - i \varepsilon_j'))$. Similarly, the line $\ell_v(b_i, \theta)$ contains the red point $r_j'$ at angle $\theta = \theta_{j-1} + \arctan((1 + i \varepsilon_j')/ (D_j + i \varepsilon_j))$. Now, for $w_j$ to cross all steps from $b_0$ to $b_n$ in order as $\theta$ increases from $\theta_{j-1}$ to $\theta_j$, the point $w_j$ must start above or on $\ell_h(b_0)$ and to the left of $\ell_v(b_0)$, which is indeed the case. Then, for $i = 0, 1, \ldots, n - 1$ the point $w_j$ must first cross $\ell_h(b_i)$, then cross $\ell_v(b_i)$, and then cross $\ell_h(b_{i+1})$. Hence, we must have...
\[ \theta_{j-1} + \arctan \left( \frac{i \varepsilon_j}{B_j - i \varepsilon'_j} \right) < \theta_{j-1} + \arctan \left( \frac{1 + i \varepsilon'_j}{D_j + i \varepsilon'_j} \right) < \theta_{j-1} + \arctan \left( \frac{(i + 1) \varepsilon_j}{B_j - (i + 1) \varepsilon'_j} \right), \]

which is equivalent to
\[ \frac{i \varepsilon_j}{B_j - i \varepsilon'_j} < \frac{1 + i \varepsilon'_j}{D_j + i \varepsilon'_j} < \frac{(i + 1) \varepsilon_j}{B_j - (i + 1) \varepsilon'_j}. \]

The last crossing must happen at the latest at angle \( \theta_j \), so we must also have
\[ \theta_{j-1} + \arctan \left( \frac{n \varepsilon_j}{B_j - n \varepsilon'_j} \right) \leq \theta_j. \quad \square \]

**Lemma 9** can be used to pick \( B_j, D_j \) such that we can create \( \Omega(n) \) blue intervals within each angular interval \([\theta_{j-1}, \theta_j)\). However, we also need to make sure that the pair \( r_j, r'_j \) actually defines the red witness during this angular interval. To show that this can be done we will show how to place the pairs \( r_j, r'_j \) one by one, that is, how to choose \( B_j, D_j \) for \( j = 1, 2, \ldots, n \).

We start with \( r_1, r'_1 \). Here we set \( B_1 = n^4 + 2n - n^2, D_1 = n^4 + n \). Note that \( \theta_0 = 0 \), so \( \varepsilon_1 = \varepsilon'_1 = 1 \). The condition from **Lemma 9** now states that we should have
\[ \frac{i}{n^4 + 2n - n^2 - i} < \frac{i + 1}{n^4 + n + i} < \frac{i + 1}{n^4 + 2n - n^2 - (i + 1)} \quad \text{for all} \ i = 0, 1, \ldots, n - 1. \]

It is easy to check that this holds for sufficiently large \( n \). Now define \( \theta_1 \) to be the angle at which \( r_1 \) leaves \( M_B(\theta) \). Thus, \( \theta_1 = \arctan(n/(n^3 + 2)) \). Then all crossings induced by the pair \( r_1, r'_1 \) happen before angle \( \theta_1 \), because (for \( n \geq 2 \)) the last condition from **Lemma 9** is satisfied:
\[ \theta_1 = \arctan \left( \frac{n}{n^3 + 2} \right) \geq \arctan \left( \frac{1}{n^3 + 1 - n} \right). \]

We also need to make sure that \( r'_1 \) is inside the blue rectangle \( M_B(\theta_1) \), which is indeed the case. In fact, \( r'_1 \) is inside \( M_B(\theta) \) for all \( 0 \leq \theta < \pi/4 \).

Now assume we have placed pairs \( r_1, r'_1 \) up to \( r_{j-1}, r'_{j-1} \) and we wish to place the pair \( r_j, r'_j \). Recall that we place \( r_j \) on the \( x_{\theta_j} \)-axis; see Fig. 6. We define \( \theta_j \) as the angle at which \( r_j \) leaves \( M_B(\theta) \).

**Observation 10.** At angles \( \theta > \theta_{j-1} \) all points \( r_1, \ldots, r_{j-1} \) are outside \( M_B(\theta) \). Hence, from \( \theta = \theta_{j-1} \) until \( \theta = \theta_j \) the point \( r_j \) is the lowest red point inside \( M_B(\theta) \).

We place \( r'_j \) on the line through the point \( b_{n^4} \) parallel to the \( x_{\theta_j} \)-axis. Thus \( r'_j \) enters \( M_B(\theta) \) at the same time that \( r_{j-1} \) leaves \( M_B(\theta) \).

**Observation 11.** At angle \( \theta = \theta_{j-1} \) the point \( r'_j \) enters \( M_B(\theta) \). When that happens, \( r'_j \) is the leftmost red point inside \( M_B(\theta) \). It remains the leftmost red point inside \( M_B(\theta) \) until \( \theta = \theta_j \), when \( r'_{j+1} \) enters \( M_B(\theta) \) and takes over as the leftmost red point.

The values \( B_j \) and \( D_j \) are chosen as follows:
\[ B_j := (n^4 + 2n) \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) - n^2, \]

and
\[ D_j := (n^4 + n) \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}). \]

To study the crossings induced by \( r_j, r'_j \), first we bound the angle \( \theta_j \) using **Lemma 12**, then in **Lemma 13** we compare \( \theta_j \) and the angles corresponding to crossings of the red witness induced by \( r_j, r'_j \) with the staircase.

**Lemma 12.** For \( n \) large enough we have \( 0 = \theta_0 < \theta_1 < \ldots < \theta_n < 2/n < \pi/12 \). Moreover, for \( 1 < i \leq n \) and \( n \) large enough we have \( \sqrt{2}/2 < \varepsilon'_i < 1 < \varepsilon_i < \sqrt{2} \).

**Proof.** It follows from the construction that \( 0 = \theta_0 < \theta_1 < \ldots < \theta_n \). To prove the first part of the lemma it thus suffices to show that \( \theta_j < 2/n \) for \( 1 \leq j \leq n \).
First, consider $\theta_1$. We have $\theta_1 = \arctan(n/(n^3 + 2))$. When $n$ tends to infinity, $n/(n^3 + 2)$ tends to zero. In this case, $\arctan(n/(n^3 + 2)) < 2n/(n^3 + 2)$. Hence, $\theta_1 < 2/n^2$. Now consider $\theta_j$ for $j > 1$. As illustrated in Fig. 7, we have $\tan(\theta_j - \theta_{j-1}) = n^2/(s + s')$, where $\sin(\pi/4 + \theta_{j-1}) = (s + s')/(\sqrt{2}(n^4 + 2n))$. Hence,

$$\theta_j = \theta_{j-1} + \arctan\left(\frac{n^2}{\sqrt{2}(n^4 + 2n) \sin(\pi/4 + \theta_{j-1})}\right).$$

Since $\sin(\pi/4 + \theta_{j-1}) > 1/2$ we have

$$\arctan\left(\frac{n^2}{\sqrt{2}(n^4 + 2n) \sin(\pi/4 + \theta_{j-1})}\right) < \arctan\left(\frac{n^2}{n^4 + 2n}\right).$$

Thus, for large value of $n$ we have

$$\theta_j < \theta_{j-1} + \arctan\left(\frac{n^2}{n^4 + 2n}\right)$$

$$< \theta_{j-1} + \frac{2n^2}{n^4 + 2n}$$

$$< \theta_1 + (j - 1) \cdot \left(\frac{2n^2}{n^4 + 2n}\right)$$

$$< 2n^2 + (j - 1) \cdot 2/n^2$$

$$\leq 2/n.$$

This proves the first part of the lemma.
To prove that for $1 < i \leq n$ and $n$ large enough we have $\sqrt{2}/2 < \epsilon'_i < 1 < \epsilon_i < \sqrt{2}$, we proceed as follows. Note that $\epsilon_i = \sqrt{2} \cdot \cos(\pi/4 - \theta_i)$ and $\epsilon'_i = \sqrt{2} \cdot \sin(\pi/4 - \theta_i)$. Further, according to the first part of the lemma we have $\theta_{i-1} < 2/n$. Hence, for $n$ large enough we have $0 < \theta_{i-1} < \pi/12$, and consequently we get $\sqrt{2}/2 < \epsilon'_i < 1 < \epsilon_i < \sqrt{6}/2 < \sqrt{2}$ which completes the proof. \qed

Next we argue that all crossings of the red witness induced by $r_j, r'_j$ with the staircase happen before angle $\theta_j$, which we can do using Lemma 9 as shown next.

**Lemma 13.** During the interval $[\theta_{j-1}, \theta_j]$ the red witness is defined by $r_j, r'_j$ and it crosses $\Theta(n)$ steps of the staircase.

**Proof.** The fact that the red witness is defined by $r_j, r'_j$ during the interval $[\theta_{j-1}, \theta_j]$ follows from Observations 10 and 11. Thus, it remains to plug the values of $B_j$ and $D_j$ into Lemma 9, and check that they satisfy the conditions. Recall that we already checked this for $j = 1$, so now assume $j > 1$.

We start with Condition (1). Define

$$E := i \epsilon_j \cdot \left( (n^4 + n) \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}) + i \epsilon_j \right),$$

and

$$E' := (1 + i \epsilon'_j) \cdot \left( (n^4 + 2n) \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) - n^2 - i \epsilon'_j \right),$$

and

$$E'' := (i + 1) \epsilon_j \cdot \left( (n^4 + n) \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}) + i \epsilon_j \right) + \epsilon'_j + i \epsilon'_j^2.$$

Then Condition (1) is equivalent to $E < E' < E''$. We first prove $E' < E''$. To this end, note that $(\cos \theta_{j-1} + \sin \theta_{j-1}) = \Theta(1)$ and $\epsilon_j, \epsilon'_j = \Theta(1)$ by Lemma 12, which implies that it suffices to consider the $n^4$-terms on both sides and prove that

$$(1 + i \epsilon'_j)n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) < (i + 1) \epsilon_j n^4 \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}).$$

This follows from Lemma 12.

Next we show that $E < E'$. Since $1 \leq i \leq n$ and $\epsilon_j, \epsilon'_j = \Theta(1)$, this means we must show

$$i \epsilon_j n^4 \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}) + O(n^3) < (1 + i \epsilon'_j)n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}). \quad (3)$$

Recall that $\epsilon_j = \sqrt{2} \cdot \cos(\pi/4 - \theta_j) = \cos \theta_j + \sin \theta_j$ and $\epsilon'_j = \sqrt{2} \cdot \sin(\pi/4 - \theta_j) = \cos \theta_j - \sin \theta_j$. Hence,

$$i \epsilon_j n^4 \cdot (\cos \theta_{j-1} + \sin \theta_{j-1}) = in^4 \cdot (\cos \theta_{j-1} + \sin \theta_{j-1})^2$$

$$= in^4 \cdot (1 + 2 \sin \theta_{j-1} \cos \theta_{j-1})$$

and

Fig. 7. Comparing $\theta_{j-1}$ and $\theta_j$. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)
(1 + iε_j)n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) = (1 + i(\cos \theta_{j-1} - \sin \theta_{j-1})) \cdot n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1})
= n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) + \epsilon_j n^4 \cdot (\cos \theta_{j-1} - \sin \theta_{j-1})^2
= n^4 \cdot \sqrt{2} \sin(\pi/4 - \theta_{j-1}) + \epsilon_j n^4 \cdot (1 - 2 \sin \theta_{j-1} \cos \theta_{j-1})

Thus, we can rewrite Inequality (3) as

\[ 4n^4 \sin \theta_{j-1} \cdot \cos \theta_{j-1} + O(n^3) < \sqrt{2}n^4 \sin(\pi/4 - \theta_{j-1}). \quad (4) \]

Next we show that Inequality (4) is satisfied for \(1 \leq i \leq n/20\), which proves that the red witness crosses the staircase \(\Theta(n)\) times. To this end, observe that since \(\theta_{j-1} < \pi/4\) we have \(\sin \theta_{j-1} < \theta_{j-1}\). Combining this with the fact that \(\theta_{j-1} < 2/n\) by Lemma 12, we get for \(i < n/20\) that

\[ 4n^4 \sin \theta_{j-1} \cdot \cos \theta_{j-1} + O(n^3) < (4/10)n^4 + O(n^3). \]

On the other hand, since \(\theta_{j-1} < \pi/12\) by Lemma 12 we have for \(n\) large enough that

\[ \sqrt{2}n^4 \sin(\pi/4 - \theta_{j-1}) > \sqrt{2}n^4 \sin(\pi/6) = \frac{1}{2} \sqrt{2}n^4. \]

We can conclude that Inequality (4) (and, hence, Condition (1) of Lemma 9) is satisfied for \(1 \leq i \leq n/20\).

It remains to prove that Condition (2) of Lemma 9 holds. (Note that we only showed Condition (1) for \(i \leq n/20\), but it follows directly from the proof of Lemma 9 that this implies that the red witness crosses the blue staircase (at least) \(n/20\) times, provided Condition (2) holds. In fact, we could replace \(n\) by \(n/20\) in Condition (2).) Now we need to prove that \(\tan(\theta_j - \theta_{j-1}) \geq (ne_j/(B_j - ne_j))\). As illustrated in Fig. 7, \(\tan(\theta_j - \theta_{j-1}) = n^2/(s + s')\), where \(s = \tan \theta_{j-1} \cdot ((n^4 + 2n) \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}))\) and \(s' = (n^4 + 2n)/\cos \theta_{j-1}\). Substituting these quantities we get

\[ \tan(\theta_j - \theta_{j-1}) = \frac{n^2 \cos \theta_{j-1}}{(n^4 + 2n) \cdot \sin \theta_{j-1} \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) + n^4 + 2n}. \]

Further,

\[ \frac{ne_j}{B_j - ne_j} = \frac{n^2 \cos \theta_{j-1}}{(n^4 + 2n) \cdot \sin \theta_{j-1} \cdot (\cos \theta_{j-1} - \sin \theta_{j-1}) + n^4 + 2n}. \]

Because \(\theta_{j-1} < 2/n\) and \(\sqrt{2}/2 < \epsilon_j < \sqrt{2}\) by Lemma 12, it is easy to see that for large value of \(n\) we have \((ne_j/(B_j - ne_j)) \leq \tan(\theta_j - \theta_{j-1})\). As a result, Condition (2) of Lemma 9 can be satisfied which completes the proof. \(\square\)

This concludes our proof that our construction produces \(\Omega(n^2)\) blue intervals as \(\theta\) increases from 0 to \(\pi/4\).

**Theorem 14.** For any \(n\) sufficiently large, there are sets \(R\) and \(B\) of total size \(n\) such that there are \(\Theta(n^2)\) disjoint angular intervals for which \(R\) and \(B\) admit a blue \(L\)-shape.

5. An output-sensitive algorithm

In the previous sections we have given an algorithm that reports all intervals for which there exists a blue \(L\)-shape in \(O(n^2)\) time, and we have seen that there are sets of points such that the number of such intervals is \(\Theta(n^2)\). Hence, our algorithm is worst-case optimal. There are situations, however, where the number of output intervals is significantly less than quadratic. Thus, in this section we propose an output-sensitive algorithm for computing the intervals for which there exists a blue \(L\)-shape. The proposed output-sensitive algorithm outperforms the worst-case optimal algorithm in terms of time complexity when the output size is subquadratic. The main idea of the proposed output-sensitive algorithm is to use the fact that the number of non-crossing events (blue-rect, blue-sc, red-set, and witness events) is small, and the number of crossing events is proportional to the output size. Thus, we will pre-compute the non-crossing events, and then compute all crossing events by using some data-structuring techniques. Recall that pre-computing all non-crossing events can be done in \(O(n \log n)\) time and \(O(nc(n))\) storage, as explained in Section 3.1.

Our output-sensitive algorithm now works as follows. After pre-computing all non-crossing events, we start our rotational sweep. During the rotational sweep, we handle each of the non-crossing events in the normal way. However, we also need to detect the crossing events, as follows.

Consider two consecutive non-crossing events, and let \(\theta_i\) and \(\theta_{i+1}\) denote the angles at which these events take place. Then we need to report all the crossing events taking place at angles \(\theta\), where \(\theta_i \leq \theta < \theta_{i+1}\). To this end, we store the staircase in a suitable data structure, as explained next. The staircase consists of horizontal edges (parallel to the \(x_0\)-axis)
and vertical edges (parallel to the $y_0$-axis). We explain how to store the horizontal edges; the vertical edges can be handled similarly. Let $p$ and $p'$ be two consecutive blue points along the staircase, where $p'$ has larger $y_0$-coordinate, and consider the horizontal staircase edge $e$ incident to $p$. Thus, the right endpoint of $e$ is $p$, and the left endpoint has the same $y_0$-coordinate as $p$ and the same $x_0$-coordinate as $p'$. Now suppose the red witness is defined by red points $r$ and $r'$, with $r'$ having larger $y_0$-coordinate. Thus, the red witness is the point $(x_0(r'), y_0(r))$. See Fig. 8(a).

This red witness crosses $e$ for some $\theta \in [\theta_i, \theta_{i+1})$ if and only if the following conditions are met: (i) the angle that $\ell(p, r)$, the line through $p$ and $r$, makes with the (original) positive $x$-axis is $\theta$, and (ii) $x_0(p') \leq x_0(r') \leq x_0(p)$. See Fig. 8(b). We now map the edge $e$ to the point $(x_0(p), y_0(p), x_0(p'), y_0(p'))$ in $\mathbb{R}^4$. Now, for any given red points $r$ and $r'$ defining the red witness and angles $\theta_i$ and $\theta_{i+1}$, there is a region $Q(r, r', \theta_i, \theta_{i+1})$ in $\mathbb{R}^4$ with the property that the red witness crosses a horizontal staircase edge $e$ if and only if $(x_0(p), y_0(p), x_0(p'), y_0(p')) \in Q(r, r', \theta_i, \theta_{i+1})$. Thus, we store the points of $Q(r, r', \theta_i, \theta_{i+1})$ in a dynamic data structure $\mathcal{D}$ so that we can perform a range query with the range $Q(r, r', \theta_i, \theta_{i+1})$. Note that $\mathcal{D}$ must be updated at each blue-sc event.

**Lemma 15.** Conditions (i) and (ii) can be expressed as a Boolean formula whose terms are polynomials in the coordinates $(x_0(p), y_0(p), x_0(p'), y_0(p'))$.

**Proof.** To prove this we focus on the interval $[0, \pi / 2)$. This can be generalized to the interval $[0, 2\pi)$ as well. Let $\theta$ be the angle that $\ell(p, r)$ makes with the positive $x$-axis. Condition (i) states that we must have $\theta \in [\theta_i, \theta_{i+1})$. In other words, we must have $\tan \theta_i \leq \tan \theta < \tan \theta_{i+1}$. Recall that $\tan \theta = \frac{y_0(r) - y_0(p)}{x_0(r) - x_0(p)}$. Substituting $\tan \theta$ in the above inequalities, Condition (i) can be expressed as conjunction of the following two inequalities:

$$y_0(r) - y_0(p) - x_0(r) \cdot \tan \theta_i + x_0(p) \cdot \tan \theta_i \geq 0$$  \hspace{1cm} (5)

and

$$x_0(r) \cdot \tan \theta_{i+1} - x_0(p) \cdot \tan \theta_{i+1} - y_0(r) + y_0(p) > 0$$  \hspace{1cm} (6)

Now for Condition (ii) we should have $x_0(p') \leq x_0(r') \leq x_0(p)$. To express these we proceed as follows. Let $\ell_0(p')$ be the line passing through $p'$, whose angle with the (original) positive $x$-axis is $\pi / 2 + \theta$. To satisfy Condition (ii), the intersection of $\ell_0(p')$ and the $x$-axis should lie to the left of the intersection of $\ell_0(r')$ and the $x$-axis. Further, the intersection of $\ell_0(r')$
and the x-axis should lie to the left of the intersection of $\ell_0(p)$ and the x-axis. The intersection of $\ell_0(p')$ and the x-axis can be written as follows: $x = y_0(p') \cdot \tan \theta + x_0(p')$. Substituting $\tan \theta$, Condition (ii) can be expressed as conjunction of the following two inequalities:

$$y_0(p') (y_0(r) - y_0(p)) + x_0(p') (x_0(r) - x_0(p)) \leq y_0(r') (y_0(r) - y_0(p)) + x_0(r') (x_0(r) - x_0(p))$$

(7)

and

$$y_0(r') (y_0(r) - y_0(p)) + x_0(r') (x_0(r) - x_0(p)) \leq y_0(p) (y_0(r) - y_0(p)) + x_0(p) (x_0(r) - x_0(p))$$

(8)

Therefore, Conditions (i) and (ii) can be expressed as conjunction of inequalities (5), (6), (7), and (8).

Hence, $Q(r, r', \theta_1, \theta_{i+1})$ is a semi-algebraic set of constant complexity. Thus, the data structure $D$ is a data structure for range searching with semi-algebraic sets in $\mathbb{R}^d$ for $d = 4$, and we can get the following performance [1,10]: for any $n \leq m \leq n^h$ and any fixed $\varepsilon > 0$, we can obtain $O(n^{1+\varepsilon}/m^{1/2} + t)$ query time (where $t$ is the number of answers) with a structure using $O(m^{1+\varepsilon})$ storage and with $O(m^{1+\varepsilon}/n)$ update time, where $b = 2d - 4 + \varepsilon = 4 + \varepsilon$. The number of queries we have to do is equal to the total number of non-crossing events, and so the number of queries is $O(n \varepsilon/n)$. We therefore set $m := n^{1/5}$. This way each query takes $O(n^{3/5+\varepsilon} + t)$ time and each update takes $O(n^{3/5+\varepsilon})$ time. After performing a query and reporting $t$ crossing events, we need to sort the crossing events in $O(t \log t)$ time to find the angular intervals we have to report. This gives an overall time for our algorithm of $O(n^{3/5+\varepsilon} + k \log k)$, where $k$ is the number of reported angular intervals. We obtain the following theorem.

**Theorem 16.** Let $B$ be a set of blue points and let $R$ be a set of red points in the plane, with $n := |B| + |R|$. Then, for any fixed $\varepsilon > 0$, we can compute all $k$ angular intervals for which there is a blue L-shape in $O(n^{3/5+\varepsilon} + k \log k)$ time and $O(n^{8/5+\varepsilon})$ storage.

6. Conclusion

We studied the following problem, which was raised by Van Kreveld et al. [12]: given a bichromatic point set in the plane of size $n$, determine all angles for which there exists an L-shape completely separating points of one color from the other. We proposed a worst-case optimal $O(n^2)$ time algorithm that uses $O(n)$ storage, and an output-sensitive algorithm that reports these angular intervals in $O(n^{8/5+\varepsilon} + k \log k)$ time and $O(n^{8/5+\varepsilon})$ storage, where $k$ is the number of reported angular intervals and $\varepsilon > 0$ is any fixed constant. One obvious open problem is to see if the running time of our output-sensitive algorithm can be improved. Another interesting problem is to develop algorithms for more general separators, for example orthogonal polygons.

**Acknowledgements**

We would like to thank Günter Rote for helpful discussions on the topic.

**References**


