Maximum-Area Rectangles in a Simple Polygon

Yujin Choi
Technische Universität Berlin, Germany
yj5162@postech.ac.kr

Seungjun Lee
Pohang University of Science and Technology, Pohang, Korea
juny2400@postech.ac.kr

Hee-Kap Ahn
Pohang University of Science and Technology, Pohang, Korea
http://tcs.postech.ac.kr/~heekap
heekap@postech.ac.kr

Abstract

We study the problem of finding maximum-area rectangles contained in a polygon in the plane. There has been a fair amount of work for this problem when the rectangles have to be axis-aligned or when the polygon is convex. We consider this problem in a simple polygon with n vertices, possibly with holes, and with no restriction on the orientation of the rectangles. We present an algorithm that computes a maximum-area rectangle in $O(n^3 \log n)$ time using $O(kn^2)$ space, where k is the number of reflex vertices of P. Our algorithm can report all maximum-area rectangles in the same time using $O(n^3)$ space. We also present a simple algorithm that finds a maximum-area rectangle contained in a convex polygon with n vertices in $O(n^3)$ time using $O(n)$ space.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Maximum-area rectangle, largest rectangle, simple polygon

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2019.12

Related Version https://arxiv.org/abs/1910.08686

Acknowledgements This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the SW Starlab support program (IITP-2017-0-00905) supervised by the IITP (Institute for Information & communications Technology Promotion).

1 Introduction

Computing a largest figure of a certain prescribed shape contained in a container is a fundamental and important optimization problem in pattern recognition, computer vision and computational geometry. There has been a fair amount of work for finding rectangles of maximum area contained in a convex polygon P with n vertices in the plane. Amenta showed that an axis-aligned rectangle of maximum area can be found in linear time by phrasing it as a convex programming problem [3]. Assuming that the vertices are given in order along the boundary of P, stored in an array or balanced binary search tree in memory, Fischer and Höffgen gave $O(\log^2 n)$-time algorithm for finding an axis-aligned rectangle of maximum area contained in P [9]. The running time was improved to $O(\log n)$ by Alt et al. [2].

Knauer et al. [12] studied a variant of the problem in which a maximum-area rectangle is not restricted to be axis-aligned while it is contained in a convex polygon. They gave randomized and deterministic approximation algorithms for the problem. Recently, Cabello et al. [6] gave an exact $O(n^3)$-time algorithm for finding a maximum-area rectangle with no restriction on its orientation that is contained in a convex n-gon. They also gave an algorithm for finding a maximum-perimeter rectangle and approximation algorithms.

39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2019).
Editors: Arkadev Chattopadhyay and Paul Gastin; Article No. 12; pp. 12:1–12:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

This problem has also been studied for containers which are not necessarily convex. Some previous work on the problem focuses on finding an axis-aligned rectangle of maximum area or of maximum perimeter contained in a rectilinear polygonal container in the plane [1, 13, 14]. Daniels et al. studied the problem of finding a maximum-area axis-aligned rectangle contained in a polygon, not necessarily convex and possibly having holes [8]. They gave $O(n \log^2 n)$-time algorithm for the problem. Later, Boland and Urrutia improved the running time by a factor of $O(\log n)$ for simple polygons with n vertices [5]. With no restriction on the orientation of the rectangles, Hall-Holt et al. gave a PTAS for finding a fat[1] rectangle of maximum area contained in a simple polygon [11].

Our results. We study the problem of finding a maximum-area rectangle with no restriction on its orientation that is contained in a simple polygon P with n vertices, possibly with holes, in the plane. We are not aware of any previous work on this problem, except a PTAS for finding a fat rectangle of maximum area inscribed in a simple polygon [11]. We present an algorithm that computes a maximum-area rectangle contained in a simple polygon with n vertices in $O(n^3 \log n)$ time using $O(kn^2)$ space, where k is the number of reflex vertices of P. Our algorithm can also find all rectangles of maximum area contained in P in the same time using (n^3) space. We also present a simple algorithm that finds a maximum-area rectangle contained in a convex polygon with n vertices in $O(n^3)$ time using $O(n)$ space.

To obtain the running time and space complexities, we characterize the maximum-area rectangles and classify them into six types, based on the sets of contacts on their boundaries with the polygon boundary. Then we find a maximum-area rectangle in each type so as to find a maximum-area rectangle contained in P. To facilitate the process, we construct a ray-shooting data structure for P of $O(n)$ space which supports, for a given query point in P and a direction, $O(\log n)$ query time. We also construct the visibility region from each vertex within P, which can be done in $O(n^2)$ time using $O(n^2)$ space in total. For some types, we compute locally maximal rectangles aligned to the coordinate axes while we rotate the axes. To do this, we maintain a few data structures such as double staircases of reflex vertices and priority queues for events during the rotation of the coordinate axes. They can be constructed and maintained in $O(kn^2 \log n)$ time using $O(kn^2)$ space. The total number of events considered by our algorithm is $O(n^3)$, each of which is handled in $O(\log n)$ time.

▶ **Theorem 1.** *We can compute a largest rectangle contained in a simple polygon with n vertices, possibly with holes, in $O(n^3 \log n)$ time using $O(kn^2)$ space, where k is the number of reflex vertices. We can report all largest rectangles in the same time using $O(n^3)$ space.*

▶ **Theorem 2.** *We can find a largest rectangle in a convex polygon P with n vertices in $O(n^3)$ time using $O(n)$ space.*

Due to lack of space, some proofs and details are omitted.

2 Preliminary

Let P be a simple polygon with n vertices in the plane. For ease of description, we assume that P has no hole. When P has holes, our algorithm works with a few additional data structures and procedures for testing if candidate rectangles contain a hole. We discuss this in Section 8. Without loss of generality, we assume that the vertices of P are given in order

[1] A rectangle is c-fat if its aspect ratio is at most c for some constant c.

along the boundary of P. We denote by k with $k \geq 1$ the number of reflex vertices of P. We assume the general position condition that no three vertices of P are on a line. Whenever we say a *largest rectangle*, it refers to a maximum-area rectangle contained in P.

We use the xy-Cartesian coordinate system and rotate the xy-axes around the origin while the polygon is stationary. We use C_θ to denote the coordinate axes obtained by rotating the xy-axes of the standard xy-Cartesian coordinate system by θ degree counterclockwise around the origin. For a point p in the plane, we use p_x and p_y to denote the x- and y-coordinates of p with respect to the coordinate axes, respectively. We say a segment or line is *horizontal* (or *vertical*) if it is parallel to the x-axis (or the y-axis). Let $\eta(p)$, $\lambda(p)$ and $\delta(p)$ denote the ray (segment) emanating from p going horizontally leftwards, rightwards and vertically downwards in the coordinate axes, respectively, until it escapes P for the first time. We call the endpoint of a ray other than its source point the *foot* of the ray. We denote the foot of $\eta(p)$, $\lambda(p)$ and $\delta(p)$ by $\bar{\eta}(p)$, $\bar{\lambda}(p)$ and $\bar{\delta}(p)$, respectively.

We use $D_\varepsilon(p)$ to denote the disk centered at a point p with radius $\varepsilon > 0$. For any two points p and q in the plane, we use pq and to denote the line segment connecting p and q, and $|pq|$ to denote the length of pq. For a segment s, we use $D(s)$ to denote the smallest disk containing s. For a subset $S \subseteq P$, we define the *visibility region* of S as $V(S) = \{x \in P \mid px \subset P \text{ for every point } p \in S\}$. For a point $p \in P$, we abuse the notation such that $V(p) = V(\{p\})$. For a set X, we use ∂X to denote the boundary of X.

2.1 Existence of a maximum-area rectangle in a simple polygon

The set $\mathcal{G}$ of all parallelograms in the plane is a metric space under the Hausdorff distance measure d_H. The Hausdorff distance between two sets A and B of points in the plane is defined as $d_H(A, B) = \max\{\sup_{a\in A} \inf_{b\in B} d(a, b), \sup_{b\in B} \inf_{a\in A} d(a, b)\}$, where $d(a, b)$ denotes the distance between a and b of the underlying metric. Since the area function $\mu : \mathcal{G} \to \mathbb{R}^{\geq 0}$ is continuous in $\mathcal{G}$, the following lemma assures the existence of a largest rectangle contained in P and thus justifies the problem. Let $\mathcal{R}$ denote the set of all rectangles contained in P. Clearly, $\mathcal{R} \subset \mathcal{G}$.

▶ **Lemma 3.** *The set $\mathcal{R}$ is compact.*

Proof. Define $f : \mathbb{R}^6 \to \mathcal{G}$ to be a function that maps a triplet (p, u, v) of points p, u, and v in $\mathbb{R}^2$ to the parallelogram in $\mathbb{R}^2$ that has p, $p + u$, $p + v$, and $p + u + v$ as the four corners.

If a parallelogram $G \in \mathcal{G}$ is not contained in P, there always exists a point $q \in G$ and a disk $D_\varepsilon(q)$ for some $\varepsilon > 0$ satisfying $D_\varepsilon(q) \cap P = \emptyset$ in the plane. Then for any parallelogram $Q \in \mathcal{G}$ with $d_H(G, Q) < \varepsilon$, the intersection $Q \cap D_\varepsilon(q)$ is not empty and Q is not contained in P. Thus, $\mathcal{C} = \{G \in \mathcal{G} \mid G \not\subset P\}$ is open in $\mathcal{G}$, and therefore $W_P = \{(p, u, v) \in \mathbb{R}^6 \mid f(p, u, v) \subset P\} = f^{-1}(\mathcal{G} \setminus \mathcal{C})$ is closed. This implies that $T_P = \{(p, u, v) \in \mathbb{R}^6 \mid f(p, u, v) \subset P, \text{a rectangle}\} = W_P \cap \{(p, u, v) \mid u \cdot v = 0\}$ is closed and also bounded in $\mathbb{R}^6$, i.e. compact. Now we can conclude that $f(T_P) = \mathcal{R}$ is also compact by f being continuous in $\mathbb{R}^6$. ◀

2.2 Classification of largest rectangles

We give a classification of largest rectangles based on the sets of contacts they have on their boundaries with the polygon boundary. We say a rectangle contained in P has a *side-contact* (sc for short) if a side has a reflex vertex of P lying on it, excluding the corners. Similarly, we say a rectangle contained in P has a *corner-contact* (cc for short) if a corner lies on an edge or a vertex of P. When two opposite corners (or two opposite sides) have corner-contacts (or side-contacts), we say the contacts are *opposite*.

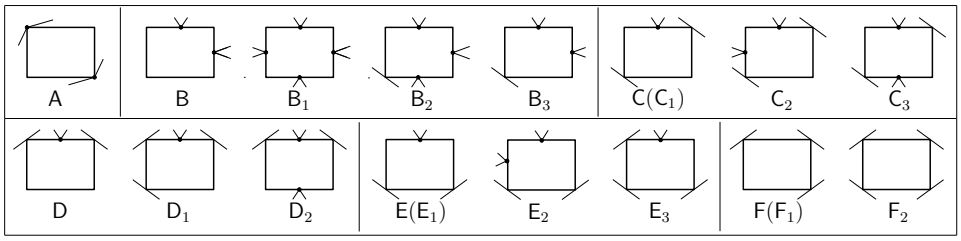

Figure 1 Classification of the determining sets of contacts of largest rectangles when rotations are allowed. Each canonical type X, except A, has a few subtypes X_i for $i = 1, 2, 3$.

Daniels et al. [8] studied this problem with the restriction that rectangles must be axis-aligned. They presented a classification of determining sets of contacts, defined below, into five types for a largest axis-aligned rectangle contained in a simple polygon in the plane.

▶ **Definition 4** (Determining set of contacts [8]). *A set Z of contacts is a determining set of contacts if the largest axis-aligned rectangle satisfying Z has finite area and the largest axis-aligned rectangle satisfying any proper subset of Z has greater or infinite area.*

In our problem, a largest rectangle R is not necessarily axis-aligned. Consider two orthogonal lines which are parallel to the sides of R and pass through the origin. Since R is aligned to the coordinate axes defined by the lines, it also has a determining set of contacts defined by Daniels et al. From this observation, we present a classification of the determining sets of contacts (DS for short) for a largest rectangle in P into six *canonical types*, from A to F, and their subtypes. The classification is given below together with the figures in Figure 1.

- Type A. Exactly two opposite ccs lying on convex vertices of P.
- Type B. One sc on each side incident to a corner c. In addition, B_1 has a sc on each of the other two sides, and B_2 and B_3 have a cc on the corner c' opposite to c. B_2 has another sc on a side incident to c'.
- Type C (C_1). Two ccs on opposite corners c and c', and a sc on a side e incident to a corner c. C_2 has another sc on the side incident to c' and adjacent to e, and C_3 has another sc on a side opposite to e.
- Type D. A sc on a side e and a cc on each endpoint of e. D_1 has another cc and D_2 has another sc on the side opposite to e.
- Type E (E_1). A sc on a side e and a cc on each endpoint of the side e' opposite to e. E_2 has another sc on a side other than e and e'. E_3 has another cc on an endpoint of e.
- Type F (F_1). ccs on three corners. F_2 has ccs on all four corners.

This classification is the same as the one by Daniels et al., except for types A, E, and F. We subdivide the last type in the classification by Daniels et al. into two types, E and F, for ease of description. A DS for type A consists of exactly two opposite corner-contacts lying on convex vertices of P while the corresponding one by Daniels et al. [8] has two opposite corners lying on the boundary (not necessarily on vertices) of P. This is because there is no restriction on the orientation of the rectangle.

2.3 Maximal and breaking configurations

Recall that $\mathcal{R}$ denotes the set of all rectangles contained in P. Our algorithm finds a largest rectangle in $\mathcal{R}$ of each (sub)type so as to find a largest rectangle contained in P. We call a rectangle that gives a *local maximum* of the area function μ among rectangles in $\mathcal{R}$ a *local*

maximum rectangle (LMR for short). We say an LMR R is of type X if R has all contacts of subtype X_i for some $i = 1, 2, \ldots$. Since a largest rectangle contained in P is a rectangle aligned to the axes that are parallel to its sides, it has contacts of at least one type defined above and is an LMR of that type. Therefore, our algorithm finds a largest rectangle among all possible LMRs of each type.

Consider a rectangle $R \in \mathcal{R}$ that satisfies a DS Z. If there is no contact other than Z, there exists a continuous transformation of R such that the transformed rectangle is a rectangle contained in P and satisfying Z. Then by such a continuous transformation the area of R may change. Imagine we continue with such a transformation until the transformed rectangle R' gets another contact. In this case, we say R' is in a *breaking configuration* (BC for short) of Z. During the transformation, the area of R' may become locally maximum. If R' is locally maximum and has no contact other than Z, we say R' is in a *maximal configuration* of Z. There can be $O(1)$ maximal configurations of Z, which can be observed from the area function of the rectangle.

▶ **Lemma 5.** *An LMR satisfying Z is in a maximal configuration or a breaking configuration.*

For a breaking configuration Z' of a DS Z, observe that $Z' \setminus Z$ is a singleton and Z' can be a BC of some other DSs. With this fact, we can classify BCs by avoiding repetition and reducing them up to symmetry. See Figure 4 for breaking configurations.

We use $\Gamma_\theta(Z)$ to denote the axis-aligned rectangle of largest area that satisfies a DS Z in C_θ. We say a DS Z is *feasible* at an orientation θ if $\Gamma_\theta(Z)$ is a rectangle contained in P. We say an orientation θ is *feasible* for Z if $\Gamma_\theta(Z)$ is contained in P.

3 Computing a largest rectangle of type A

It suffices to check all possible squares in P with two opposite corners on convex vertices of P. Since ∂P is a simple closed curve, we can determine if a rectangle R is contained in P by checking if all four sides of R are contained in P.

▶ **Lemma 6.** *We can compute a largest rectangle among all LMRs of type A in $O((n-k)n + (n-k)^2 \log n)$ time using $O(n)$ space.*

4 Computing a largest rectangle of type B

We show how to compute all LMRs of type B and a largest rectangle among them. We compute for each DS a largest LMR over the maximal and breaking configurations. In doing so, we maintain a combinatorial structure for each reflex vertex which helps compute all LMRs of type B during the rotation of the coordinate axes.

4.1 Staircase of a point in a simple polygon

We define the staircase $\mathsf{S}(u)$ of a point $u \in P$ as the set of points $p \in P$ with $p_x \leq u_x$ and $p_y \leq u_y$ such that the axis-aligned rectangle with diagonal up is contained in P but no axis-aligned rectangle with diagonal uq is contained in P for any point $q \in P$ with $q_x < p_x$ and $q_y < p_y$. Thus, $\mathsf{S}(u)$ can be represented as a chain of segments. See Figure 2 (a).

The staircase of a point $u \in P$ in orientation θ, denoted by $\mathsf{S}_\theta(u)$, is defined as the staircase of u in C_θ. Every axis-aligned segment of $\mathsf{S}_\theta(u)$ has one endpoint at a vertex of P, $\bar{\eta}(u)$, or $\bar{\delta}(u)$. A segment of $\mathsf{S}_\theta(u)$ that is not aligned to the axes is a part of an edge e of P and is called an *oblique segment*. (We say e appears to the staircase in this case.)

Each vertex of $\mathsf{S}_\theta(u)$ which is a vertex P, $\bar{\eta}(u)$, or $\bar{\delta}(u)$ is called an *extremal vertex*. An extremal vertex v is called a *tip* if it is a reflex vertex of P. A vertex of $\mathsf{S}_\theta(u)$ contained in P is called a *hinge*. A *step* of $\mathsf{S}_\theta(u)$, denoted by an ordered pair (a, b), is the part of $\mathsf{S}_\theta(u)$ between two consecutive extremal vertices a and b along $\mathsf{S}_\theta(u)$, where $a_x \leq b_x$ and $a_y \geq b_y$. It consists of either (i) two consecutive segments ar and rb for $r = \delta(a) \cap \eta(b)$, or (ii) three consecutive segments $a\bar{\delta}(a)$, $\bar{\delta}(a)\bar{\eta}(b)$, and $\bar{\eta}(b)b$ with $\bar{\delta}(a)_x \leq \bar{\eta}(b)_x$ and $\bar{\delta}(a)_y \geq \bar{\eta}(b)_y$. Note that $\bar{\delta}(a)\bar{\eta}(b)$ is the oblique segment of step (a, b) which we denote by $\mathsf{ob}(a, b)$. A horizontal, vertical, or oblique segment of a step can be just a point in case of degeneracy.

We can construct $\mathsf{S}_\theta(u)$ for a fixed θ in $O(n)$ time by traversing the boundary of P in counterclockwise direction starting from $\bar{\eta}(u)$ while maintaining the staircase of u with respect to the boundary chain traversed so far. When the next vertex v of the boundary chain satisfies $v_x \geq t_x$ and $v_y \leq t_y$ for the last vertex t of the current staircase, we append it to the staircase. If (part of) the edge incident to v is an oblique segment of the staircase, then we append it together with v to the staircase. If $v_x < t_x$, we ignore v and proceed to the vertex next to v. If $v_x \geq t_x$ and $v_y > t_y$, we remove the portion of the current staircase violated by v and append v to the staircase accordingly. Observe that each vertex and each edge appear on the staircase at most once during the construction.

4.2 Maintaining the staircase during rotation of the coordinate system

Bae et al. [4] considered the rectilinear convex hull for a set Q of n point in the plane and presented a method of maintaining it while rotating the coordinate system in $O(n^2)$ time. The boundary of the rectilinear convex hull consists of four maximal chains, each of which is monotone to the coordinate axes. We adopt their method and maintain the staircase of a reflex vertex u in a simple polygon.

The combinatorial structure of $\mathsf{S}_\theta(u)$ changes during the rotation. Figure 2 (b-f) show $\mathsf{S}_0(u)$, $\mathsf{S}_{\theta_1}(u)$ and $\mathsf{S}_{\theta_2}(u)$ for three orientations $0, \theta_1$, and θ_2 ($0 < \theta_1 < \theta_2 < \pi/2$). Two consecutive steps, (a, b) and (b, c), of the staircase merge into one step (a, c) when $\bar{\delta}(a)$ meets b (Figure 2 (b)). A step (a, b) splits up into two steps (a, v) and (v, b) when $\bar{\eta}(b)$ meets a polygon vertex v (Figure 2 (c)). A step changes its type between (A) and (B) when the hinge of a step hits a polygon edge (and then it is replaced by an oblique segment) or the oblique segment of a step degenerates to a point (and then it becomes a hinge) (Figure 2 (d)). The upper tip a (or the lower tip b) of a step (a, b) can disappear from the staircase when $\bar{\delta}(\bar{\eta}(u))$ meets a (or $\bar{\eta}(\bar{\delta}(u))$ meets b). Finally, a vertex, possibly along with an edge incident to it, can be added to or deleted from $\mathsf{S}_\theta(u)$ when it is met by $\bar{\eta}(\bar{\delta}(u))$ or $\bar{\delta}(\bar{\eta}(u))$. We call such a change of the staircase due to the cases described above a *step event*.

One difference of the staircase $\mathsf{S}_\theta(u)$ to the one for a point-set by Bae et al. is that the two boundary points of $\mathsf{S}_\theta(u)$ are $\bar{\eta}(u)$ and $\bar{\delta}(u)$. Since the polygon is not necessarily monotone with respect to the axes, the staircase may change discontinuously when $\bar{\eta}(u)$ or $\bar{\delta}(u)$ meets a vertex of P, which we call a *ray event*. The step of $\mathsf{S}_\theta(u)$ incident to $\bar{\eta}(u)$ is replaced by a chain of $O(n)$ steps when $\bar{\eta}(u)$ meets a vertex of P (Figure 2 (e)). A subchain incident to $\bar{\delta}(u)$ is replaced by a single step when $\bar{\delta}(u)$ meets a vertex of P (Figure 2 (f)). We call the appearance or disappearance of a step caused by a ray event a *shift event* of the ray event. Note that $O(n)$ shift events occur at a ray event. Observe that all the changes occurring in a staircase during the rotation are caused by step, ray, or shift events. We abuse $\mathsf{S}_\theta(u)$ to denote the combinatorial structure of the staircase if understood in context.

▶ **Lemma 7.** *The number of events that occur to $\mathsf{S}_\theta(u)$ during the rotation is $O(n^2)$.*

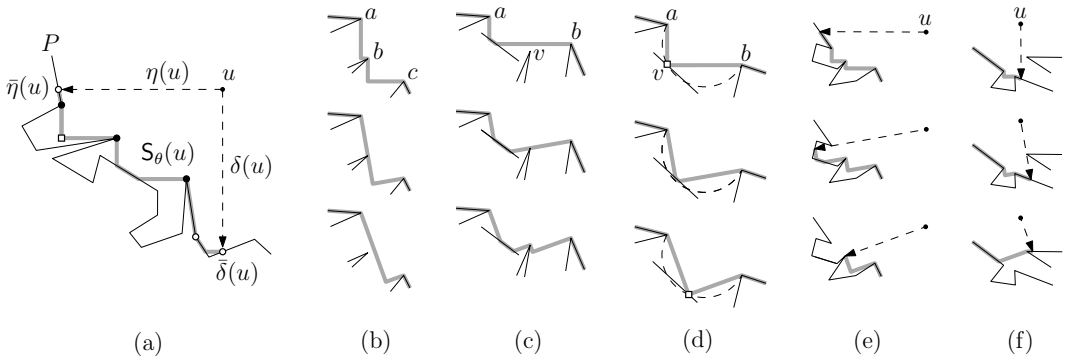

Figure 2 (a) Staircase $S_\theta(u)$ (thick gray chain) and the tips (black disks), the extremal vertices (black disks and circles), and the hinge (square) of $S_\theta(u)$. (b–d) Step events, and (e–f) ray events during the rotation of the coordinate system.

To capture these combinatorial changes and maintain the staircase during the rotation, we construct for every reflex vertex p of P, the list of segments of visibility region $V(p)$ sorted in angular order. We compute for every pair (p, q) of reflex vertices of P, the list $C(p, q)$ of vertices and segments of $\partial V(\{p, q\}) \cap D(pq)$, sorted in angular order with respect to p and q. We also compute for every pair (p, e) of a reflex vertex p and edge e, the sorted list $L(p, e)$ of angles at which $\delta(\bar{\eta}(p))$ or $\eta(\bar{\delta}(p))$ meets a vertex of P while $\bar{\eta}(p)$ or $\bar{\delta}(p)$ lies on e. We store for each orientation in $L(p, e)$ the information on the vertex corresponding to the orientation. This can be computed by finding the points that e intersects with the boundary of $D(tp)$ for each vertex t of P. These structures together constitute the *event map*.

We also construct an *event queue* for each reflex vertex, which is a priority queue that stores events indexed by their orientations. This is to update the staircase during the rotation in a way similar to the one by Bae et al. [4] using the event map. The event map is of size $O(kn^2)$ and can be constructed in $O(kn^2 \log n)$ time. For a reflex vertex u, we maintain $\mathsf{S}_\theta(u)$ and the event queue $\mathcal{Q}$ for u during the rotation using the event map. We also store the extremal vertices and edges of the staircase in a balanced binary search tree $\mathcal{T}$ representing $\mathsf{S}_\theta(u)$ in order along the staircase so as to insert and delete an element in $O(\log n)$ time.

▶ **Lemma 8.** *The staircases of all k reflex vertices of P can be constructed and maintained in $O(kn^2 \log n)$ time using $O(kn^2)$ space during the rotation.*

4.3 Data structures – double staircases, event map, and event queue

Our algorithm computes all LMRs of type B during the rotation and returns an LMR with largest area among them. To do so, it maintains for each reflex vertex u two staircases, $S_\theta(u)$ and $S_{\theta+\frac{\pi}{2}}(u)$ which we call the *double staircase* of u, during the rotation of the coordinate axes and computes the LMRs of type B that have u as the top sc. Let I denote the interval of orientations such that the horizontal line with respect to any $\theta \in I$ passing through u is tangent to the boundary of P locally at u. Let R_θ be the largest axis-aligned rectangle of type B in $\theta \in I$ that is contained in P and has u as the top sc. Observe that every reflex vertex lying on the right side is a tip of $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. We use X to denote the contact set around the bottom-left corner c_θ of R_θ. Then X contains (1) a tip of $\mathsf{S}_\theta(u)$ touching the left side and a tip on either $\mathsf{S}_\theta(u)$ or $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ touching the bottom side (type B_1), (2) an oblique segment e on $\mathsf{S}_\theta(u)$ touching c_θ and a tip on either $\mathsf{S}_\theta(u)$ or $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ touching the bottom side (type B_2), or (3) just an oblique segment e on $\mathsf{S}_\theta(u)$ touching c_θ (type B_3).

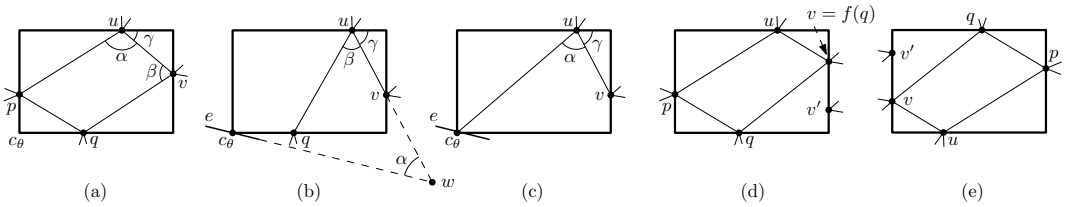

Figure 3 LMRs of (a) type B_1, (b) type B_2, and (c) type B_3. (d) Step event on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ that $f(q)$ changes from v to v'. (e) The step event in (d) is a step event on $\mathsf{S}_{\theta+\pi}(q)$.

For a reflex vertex u of P, we construct the double staircase of u, $\mathsf{S}_0(u)$ and $\mathsf{S}_{\frac{\pi}{2}}(u)$. Then we maintain the event queue $\mathcal{Q}$ containing event orientations in order: the orientations for staircase events (step and ray events) defined in previous section and the orientations at which two vertices of $V(u)$ are aligned horizontally. The set of the orientations of the latter type is to capture the event orientations at which a tip of $\mathsf{S}_\theta(u)$ is aligned horizontally with a tip of $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. We call them *double staircase events*. We initialize $\mathcal{Q}$ with the latter type events. Note that it does not increase the time and space complexities of the event queue.

4.4 Computing LMRs of type B_1

Consider a reflex vertex t of P that appears as a tip on $\mathsf{S}_\theta(u)$. We use $f(t)$ to denote the upper tip of the step on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ aligned horizontally to t. For example, in Figure 3(a,b), $v = f(q)$ in $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. During the rotation of the coordinate axes, we consider the change of $f(t)$ for each tip t on $\mathsf{S}_\theta(u)$, as well as step, ray, and shift events on the double staircase. At each orientation, $f(t)$ can be computed in $O(\log n)$ time via binary search on $S_{\theta+\frac{\pi}{2}}(u)$ with t_y since the staircase chain is monotone with respect to the y-axis. Thus we do not need to save the value $f(t)$ for each tip t. We consider the orientation when a tip t on $\mathsf{S}_\theta(u)$ and $f(t)$ on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ are aligned horizontally so that $f(t)$ is set to the next tip on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. At such a orientation, we detect a DS candidate of the type B_1 with top, left, bottom, and right scs as u, p, q, and $v = f(q)$, respectively. Note that the bottom sc q might have $q_x > u_x$. We process only the case that q is a tip on $\mathsf{S}_\theta(u)$, since the other case can be handled when fixing p as an upper side contact, as described in the following, when step (q, v) disappears by a step event on $\mathsf{S}_{\theta+\frac{\pi}{2}}(p)$ and u is a tip on $\mathsf{S}_{\theta+\pi}(p)$.

Consider an event E occurring at θ. When a step of the double staircase that possibly contributes to a DS of type B_1 changes due to E, we detect possible DSs that have been associated with it. Consider a DS $\{u, p, q, v\}$ as in Figure 3(a). If E is a step or shift event on $\mathsf{S}_\theta(u)$, $O(1)$ tips appear or disappear on $\mathsf{S}_\theta(u)$ and $O(1)$ DSs are detected at each such event. If E is a step or shift event on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$, there are $O(n)$ tips t on $\mathsf{S}_\theta(u)$ such that $f(t)$ changes. Observe that such an event corresponds to a step event on the staircase of q in $\mathsf{C}_{\theta+\pi}$. See Figure 3(d-e). Thus, we may consider only step and shift events on $\mathsf{S}_\theta(u)$ together with the double staircase events to detect possible DSs of type B_1.

We consider $O(1)$ DSs for each event. Let $Z = \{u, p, q, v\}$ be a DS of type B_1 such that (p, q) is changed or q and v are aligned horizontally by an event E at θ_E. Given a closed interval J, we can compute the set Θ_Z of orientations $\theta_Z \in J$ maximizing $\mu(\Gamma_\theta(Z))$ locally in $O(1)$ time because the area function $\mu(\Gamma_\theta(Z)) = (|uv|\cos\gamma + |up|\cos(\pi - (\alpha+\gamma)))(|uv|\sin\gamma + |qv|\cos(\frac{\pi}{2} - (\beta - \gamma)))$ has $O(1)$ extremal values in J. For angles α, β, γ, see Figure 3(a).

We find the maximal interval J of orientations for $Z = \{u, p, q, v\}$ found in an event E occurring at θ_E such that $\theta_E \in J$ and all elements of Z appear on the double staircase of u. This can be done by maintaining the latest orientation ($< \theta_E$) at which (p, q) starts to

appear as a step, the latest orientation ($< \theta_E$) at which v starts to appear as a tip to the double staircase, and the orientation at which $f(q)$ was set to v. Then $J = [\theta_a, \theta_E]$, where θ_a is the latest of orientation at which all elements of Z and the step consisting of elements of Z start to appear on the double staircase while satisfying $f(q) = v$. Note that the LMRs with contact Z occur at every orientation of Θ_Z and the two endpoints (orientations) of J. Observe that the rectangle $\Gamma_{\theta_Z}(Z)$ with $\theta_Z \in \Theta_Z$ corresponds to a maximal configuration, and $\Gamma_\theta(Z)$ with θ being an endpoint of J corresponds to a breaking configuration. In this way, we can compute $O(1)$ LMRs satisfying Z in $O(1)$ time.

For a reflex vertex u, we maintain an event queue $\mathcal{Q}$. For each event E in $\mathcal{Q}$, our algorithm finds $O(1)$ DSs Z that become infeasible by E, and computes the LMRs in $O(1)$ time. Observe that a DS Z of type B_1 becomes infeasible only at shift, step, and double staircase events. Since our algorithm is applied to every reflex vertex u of P, we do not need to process the shift and step events occurring on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. Therefore, we can detect every possible DS Z by processing the events in Q. By Lemma 8, we have the following lemma.

▶ **Lemma 9.** *Our algorithm computes all LMRs of type B_1 with largest area in $O(kn^2 \log n)$ time, where k is the number of reflex vertices.*

4.5 Computing LMRs of type B_2

A DS $Z = \{u, e, q, v\}$ of type B_2 consists of three reflex vertices u, q, v realizing the top, bottom, right sc and an oblique segment e realizing the cc at the bottom-left corner c_θ of $\Gamma_\theta(Z)$. Let w be the point where the extended line of e and the line through u and v cross. If w appears below c_θ, then the area function is $\mu(\Gamma_\theta(Z)) = |uq| \sin(\beta+\gamma)\big(\cot(\gamma - \alpha)(|uw| \sin\gamma - |uq| \sin(\beta+\gamma)) - |vw| \cos\gamma\big)$. See Figure 3(b). The area of $\Gamma_\theta(Z)$ with w appearing above u can be computed in a similar way. Note that there are $O(1)$ orientations that maximize $\mu(\Gamma_\theta(Z))$ locally and they can be computed in $O(1)$ time.

Observe that q is contained in $\mathsf{S}_\theta(u)$ or (q, v) is a step on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. In addition to the method from the Section 4.4, we also handle the case that (q, v) is a step on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ as follows. For a reflex vertex t appearing as a tip on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$, let $g(t)$ be the edge that contains $\bar{\eta}(t)$. We consider every change of $f(t)$ for each tip t on $\mathsf{S}_\theta(u)$ and the every change of $g(t)$ for each tip t on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ during the rotation.

Consider an event E occurring at θ_E. If $f(q)$ changes from v to v', we detect the DS $\{u, e, q, v\}$, where e is the edge containing the oblique segment of the step q belongs to, in a similar way as we process such an event of type B_1.

So it remains to consider the case for an event E that changes $g(q)$ for each tip q on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ at θ_E. Observe that $g(q)$ changes only if a double staircase event occurs associated with q. Whenever a new step (q, v) appears on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$, we do binary search for $\bar{\eta}(q) \in e$ in $V(q)$. When $g(q)$ changes or a step (q, v) disappears caused by a step or a shift event on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$, we find $O(1)$ LMRs with DS $Z = \{u, e, q, v\}$, and check if they are in P by checking if the boundary of the rectangles are in P, since we do not know if e appears on $\mathsf{S}_\theta(u)$ or not. These LMRs can be computed in a similar way as we do for type B_1. Together with Lemma 8, we have the following lemma.

▶ **Lemma 10.** *Our algorithm computes all LMRs of B_2 with the largest area in $O(kn^2 \log n)$ time, where k is the number of reflex vertices of P.*

4.6 Computing LMRs of type B_3

Consider the case when DS $Z = \{u, e, v\}$ is feasible, where e is a polygon edge that appears as an oblique segment on $\mathsf{S}_\theta(u)$ and v is a tip on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ (type B_3). See Figure 3(c). To achieve the largest area, we observe that the bottom-left corner of LMRs satisfying Z must lie at the midpoint c of the extended line segment pq of e, where p and q are the intersection points of the line containing e with $\eta(u)$ and $\delta(v)$, respectively. The area function of $\Gamma_\theta(\{u, c_\theta, v\})$, the rectangle with top sc on u, bottom-left cc on c_θ, and right sc on v, is convex with respect to $c_\theta \in l$, where l is the line containing e. If c does not lie on e, c_θ must lie on a point of e closest to c to maximize the rectangle area.

The area function of $\Gamma_\theta(Z)$ for a DS Z of B_3 is $\mu(\Gamma_\theta(Z)) = |uc_\theta| \sin(\alpha+\gamma)\big(|uc_\theta| \cos(\pi - (\alpha+\gamma)) + |uv| \cos\gamma\big)$, where c_θ is the midpoint of pq (if the midpoint lies on e) or the endpoint of e that is closer to the midpoint (otherwise) at θ. Since the midpoint moves along l in one direction as θ increases, there are $O(1)$ intervals of orientations at which the midpoint of pq is contained in e, and thus this area function has $O(1)$ extremal values in I.

Our algorithm for computing all LMRs of type B_3 is simple. First we fix the top sc on u. For each pair of an edge e and a reflex vertex v, we compute the set Θ_Z of orientations that maximize $\mu(\Gamma_\theta(Z))$ locally for $Z = \{u, e, v\}$. Observe that Θ_Z consists of $O(1)$ orientations because the area function has $O(1)$ extremal values. Then for each $\theta_Z \in \Theta_Z$, we find two orientations θ_1, θ_2 closest to θ_Z with $\theta_1 \le \theta_Z$ and $\theta_2 \ge \theta_Z$ such that the top-right corner of $\Gamma_\theta(Z)$ is contained in P by applying binary searching on $C(u, v)$. Finally, we check if $\Gamma_\theta(Z)$ is contained in P for $O(1)$ such orientations θ by checking if the boundary of the rectangles are contained in P. This way we can compute all LMRs of B_3 with top sc on u. See Figure 4. By using the event map and Lemma 8, we have the following lemma.

▶ **Lemma 11.** *Our algorithm computes all LMRs of B_3 with largest area in $O(kn^2 \log n)$ time, where k is the number of reflex vertices of P.*

5 Computing a largest rectangle of types C and D

LMRs of types C and D can be computed in a way similar to the one for type B. For each reflex vertex u, we find all LMRs of types C and D that have u on its top side while maintaining the double staircase of u.

▶ **Lemma 12.** *We can compute a largest rectangle among all LMRs of types C and D in $O(kn^2 \log n)$ time using $O(kn^2)$ space, where k is the number of reflex vertices of P.*

6 Computing a largest rectangle of type E

We consider the LMRs of type E. Let u be a reflex vertex of P. We detect every DS Z of type E, containing $\{u, e_l, e_r\}$, where u is the top sc, e_l the bottom-left cc, and e_r the bottom-right cc. Observe that for each LMR satisfying Z, e_l and e_r appear as oblique segments $\mathsf{ob}(p, q) \subset e_l$ and $\mathsf{ob}(t, v) \subset e_r$ of $\mathsf{S}_\theta(u)$ and $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$, respectively, such that $\bar{\eta}(t) \in \mathsf{ob}(p, q)$ or $\bar{\lambda}(q) \in \mathsf{ob}(t, v)$, depending on whether $q_y \le t_y$ or not. Using this fact, we detect the events at which Z becomes infeasible, and compute the LMRs satisfying Z.

We compute LMRs of type E at (1) every step and shift event (and ray event) with a step containing an oblique segment on the double staircase, and (2) every event such that $\bar{\lambda}(q)$ meets $\bar{\delta}(t)$ on an edge e for a tip q of $\mathsf{S}_\theta(u)$ and a tip t of $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$. At an event E of case (1), we find DSs Z that become infeasible caused by E. At an event E of case (2) occurring at θ_E, we have a step (p, q) on $\mathsf{S}_\theta(u)$ and a step (t, v) on $\mathsf{S}_{\theta+\frac{\pi}{2}}(u)$ such that $\bar{\lambda}(q)$ meets $\bar{\delta}(v)$

on an edge of P. We consider the same DSs Z_1 and Z_2 considered in case (1). Observe that E corresponds to the step event of the double staircase of v at $\theta_E - \frac{\pi}{2}$. The double staircase of v has a step event at $\theta_E - \frac{\pi}{2}$ that $\bar{\delta}(\bar{\lambda}(v))$ meets q (equivalently, $\bar{\lambda}(q)$ meets $\bar{\delta}(v)$ on an edge of P at θ_E). Thus, we can capture E by maintaining the double staircase of v and insert E to the event queue of u in $O(\log n)$ time. We compute this type of events for all reflex vertices of P by maintaining double staircases of the reflex vertices of P and insert the events to the event queues of their corresponding reflex vertices whenever such events are found. There are $O(kn^2)$ events in total, and they can be found and inserted to the event queues in $O(kn^2 \log n)$ time.

Whenever detecting a DS Z, we take a closed interval J of orientations at which Z is possibly feasible, and compute $O(1)$ LMRs with contact Z within J. When Z contains $\{u, e_l, e_r\}$ as the top sc u, bottom-left cc e_l, and bottom-right cc e_r, J is the interval such that $\mathsf{ob}(p, q) \subseteq e_l$, $\mathsf{ob}(t, v) \subseteq e_r$, and $\bar{\lambda}(\bar{\delta}(p)) \in \mathsf{ob}(t, v)$ or $\bar{\lambda}(q) \in \mathsf{ob}(t, v)$. Note that the interval satisfying $\bar{\lambda}(\bar{\delta}(p)) \in \mathsf{ob}(t, v)$ or $\bar{\lambda}(q) \in \mathsf{ob}(t, v)$ can be computed in $O(\log n)$ time using binary search on $L(p, e_l)$ and $L(v, e_r)$. If Z contains p as the left sc or e' as the top-left cc, we consider the BCs such that v is the right sc or there is another cc on the top-right corner. Note that the BC of the second case corresponds to a BC of type D_1. The orientation at which such a BC occurs can be computed in $O(1)$ time by solving basic system of linear equations. Therefore, J can be computed in $O(\log n)$ time.

We compute $O(1)$ LMRs for each event and check if they are contained in P in $O(\log n)$ time. There are $O(kn^2)$ events corresponding to case (2) which are computed in $O(kn^2 \log n)$ time before we handle the events of type E. By Lemma 7, we have the following lemma.

▶ **Lemma 13.** *We can compute a largest rectangle among all LMRs of type* E *in* $O(kn^2 \log n)$ *time using* $O(kn^2)$ *space, where* k *is the number of reflex vertices of* P*.*

7 Computing a largest rectangle of type F

To find all LMRs of type F, we compute the maximal configurations and breaking configurations of DSs of type F as follows. Consider a DS $Z_1 = \{e_1, e_l, e_2\}$ of type F_1 and a DS $Z_2 = \{e_1, e_l, e_r, e_2\}$ of type F_2. Then the LMR of a BC of type F_1 is the rectangle satisfying $Z_1 \cup \{u\}$ or $Z_1 \cup \{e_r\}$, or a rectangle satisfying Z_1 with cc on an end vertex of an edge in Z_1, where u is a reflex vertex and e_r is an edge of P. The LMR satisfying $Z_1 \cup \{u\}$ belongs to type D_1 or E_3 which is computed as an LMR of D or E. The LMR satisfying $Z_1 \cup \{e_r\}$ belongs to type F_2 and it is considered for type F_2. The LMR of a BC of type F_2 is the rectangle satisfying $Z_2 \cup \{u\}$, $Z_2 \cup \{e'\}$ or the rectangle satisfying Z_2 with cc on an end vertex of an edge in Z_2, where u is a reflex vertex and e' is an edge of P. The LMR satisfying $Z_2 \cup \{u\}$ belongs to a BC of type E_3 which is computed as an LMR of type E. (See the last BC of type E_3 in Figure 4.) Thus, we consider the LMRs of maximal configurations of type F or breaking configurations Z of type F containing a cc on an end vertex of an edge in Z only.

We say an edge pair (e_1, e_2) is *h-aligned* (and *v-aligned*) at θ if there are points $p_1 \in e_1$ and $p_2 \in e_2$ such that p_1p_2 is horizontal (and vertical) and is contained in P at θ. A pair (e_1, e_2) of edges is *h-misaligned* (and *v-misaligned*) at θ if the pair is not h-aligned (and not v-aligned) at θ. Note that a edge pair (e_1, e_2) changes between being h- or v-aligned and being h- or v-misaligned only when two vertices of P are aligned horizontally or vertically during the rotation. We say a triplet (e_1, e_l, e_2) of edges *t-aligned* at θ if there is a point $x \in e_1$ such that $\bar{\lambda}(x) \in e_2$ and $\bar{\delta}(x) \in e_l$ at some $\theta' \in \{\theta, \theta + \frac{\pi}{2}, \theta + \pi, \theta + \frac{3\pi}{2}\}$. An edge triplet (e_1, e_l, e_2) is *t-misaligned* if it is not t-aligned at θ.

We compute LMRs of type F at (1) every event such that two vertices are aligned horizontally or vertically, and (2) every event such that $\bar{\delta}(\bar{\eta}(u))$ meets p for every vertex pair (u, p) at $\theta' \in \{\theta, \theta + \frac{\pi}{2}, \theta + \pi, \theta + \frac{3\pi}{2}\}$. In case (1), at an event such that two vertices u and v are aligned horizontally, we find an edge pair (e_1, e_2) which becomes h-misaligned in $O(\log n)$ time using ray-shooting queries with $\eta(u)$ and $\lambda(v)$, assuming that $u_x < v_x$ if such pair exists. Then we also find edges e_l and e_r such that e_l contains $\bar{\delta}(\bar{\eta}(u))$ and e_r contains $\bar{\delta}(\bar{\lambda}(v))$. We can find such edges in $O(\log n)$ time using ray-shooting queries. Then we compute the set Θ_{Z_i} of orientations that maximize $\mu(\Gamma_\theta(Z_i))$ for each DS $Z_1 = \{e_1, e_l, e_2\}$, $Z_2 = \{e_1, e_r, e_2\}$ and $Z_3 = \{e_1, e_l, e_r, e_2\}$ and check if $\Gamma_\theta(Z_i)$ is contained in P for $\theta \in \Theta_{Z_i}$. Observe that the top-left cc of every LMR of type F_1 lies at the midpoint c of wt, for the intersection w of two lines, one containing e_1 and one containing e_l, and the intersection t of two lines, one containing e_1 and one containing e_2. Note that every area function of type F as $O(1)$ extremal values. If $c \notin e_1$, we take the point on e_1 that is closest to the midpoint. Thus, each Θ_{Z_i} has $O(1)$ elements and we can check for each $\Gamma_\theta(Z_i)$ if it is contained in P in $O(\log n)$ using ray-shooting queries. We also compute the BCs satisfying Z_i with cc on an end vertex of an edge in Z_i, and check their feasibility. There are $O(1)$ such BCs which can be computed in $O(1)$ time. We can compute in $O(1)$ time $\mu(\Gamma_\theta(Z))$ for each BC Z. An event at which two vertices u and v are aligned vertically can be handled in a symmetric way.

In case (2), when $\bar{\delta}(\bar{\eta}(u))$ meets p, we find an edge triplet (e_1, e_l, e_2) which becomes t-misaligned in $O(\log n)$ time using ray-shooting queries with $\eta(u), \lambda(u)$ and $\delta(p)$. We also find edges e_r and e'_r such that $\bar{\delta}(\bar{\lambda}(u)) \in e_r$ and $\bar{\lambda}(p) \in e'_r$ in $O(\log n)$ time using ray-shooting queries. Similar to case (1), we compute Θ_{Z_i} for each DS $Z_1 = \{e_1, e_l, e_2\}$, $Z_2 = \{e_1, e_l, e_r, e_2\}$ and $Z_3 = \{e_1, e_l, e'_r, e_2\}$ and check if $\Gamma_\theta(Z_i) \subseteq P$ for $\theta \in \Theta_{Z_i}$. Then we compute the BCs satisfying Z_i with cc on an end vertex of an edge in Z_i and check their feasibility.

There are $O(n^2)$ events corresponding to case (1) and $O(n^3)$ events corresponding to case (2). We can compute them in $O(n^3)$ time in total. For each event, we find $O(1)$ DSs in $O(\log n)$ time and compute $O(1)$ maximal and breaking configurations of each DS in $O(1)$ time, and check their feasibility in $O(\log n)$ time. And the only data structure we use for type F is a ray-shooting data structure of $O(n)$ space.

▶ **Lemma 14.** *We can compute a largest rectangle among all LMRs of type F in $O(n^3 \log n)$ time using $O(n)$ space.*

8 Computing a largest rectangle in a simple polygon with holes

Our algorithm can compute a largest rectangle in a simple polygon P with h holes and n vertices. We use the same classification of largest rectangles and find the LMRs of the six types. We construct a ray-shooting data structure, such as the one by Chen and Wang [7] in $O(n + h^2 \operatorname{polylog} h)$ time using $O(n + h^2)$ space, which supports a ray-shooting query in $O(\log n)$ time. We also construct the visibility region from each vertex of P, which can be done in $O(n^2 \log n)$ time using $O(n^2)$ space by using the algorithm in [7]. Each visibility region is simple and has $O(n)$ complexity. The staircase of a vertex of P can be constructed in $O(n \log n)$ time using plane sweep with ray-shooting queries. Each staircase of a vertex u of P has $O(n)$ space. There are $O(n^2)$ events to the staircase of u since it is equivalent to the staircase constructed in $V(u)$, a simple polygon with $O(n)$ vertices.

We say a rectangle is *empty* if there is no hole contained in it. Since P has holes, there can be a hole contained in a rectangle R even though every side of R is contained in P. Thus, we check the emptiness of rectangles, together with the test for their sides being contained in P. The emptiness of a rectangle can be checked by constructing a triangular

range searching data structure for n vertices of P in $O(n^2)$ time and space [10]. For a query with two triangles obtained from subdividing the rectangle by a diagonal, it answers the number of vertices lying in the triangle in $O(\log n)$ time. Since the remaining part of our algorithm works as it is, we have Theorem 1.

9 Computing a largest rectangle in a convex polygon

When P is convex, there is no reflex vertex and therefore it suffices to consider only the LMRs of types A and F. Using the method in Lemma 6, we can compute a largest LMR of type A in $O(n^2 \log n)$ time using $O(n)$ space.

For type F, we find the events considered in Section 7 and all DSs corresponding to the events in case (1) that a vertex u is aligned to another vertex in $O(n)$ time by maintaining rays $\lambda(u)$, $\delta(\bar{\lambda}(u))$, $\delta(u)$ and $\lambda(\bar{\delta}(u))$ during the rotation. Since P is convex, the foot of each ray emanating from u changes *continuously* along the boundary of P. Similarly, we find all DSs corresponding to the events in case (2) in Section 7 that an edge triplet becomes t-misaligned. It is caused by $\bar{\delta}(\bar{\eta}(u))$ meeting p for a vertex pair (u, p) and its corresponding DSs can be computed in $O(n)$ time by maintaining $\eta(u)$, $\delta(u)$, $\xi(p)$ and $\lambda(p)$ during the rotation, where $\xi(p)$ is the vertically upward ray from p. Thus, we can find all events and their corresponding DSs in $O(n^2)$ time for case (1) and in $O(n^3)$ time for case (2). Since every LMR is contained in P, we can find the maximal configuration of each DS in $O(1)$ time. We conclude with Theorem 2.

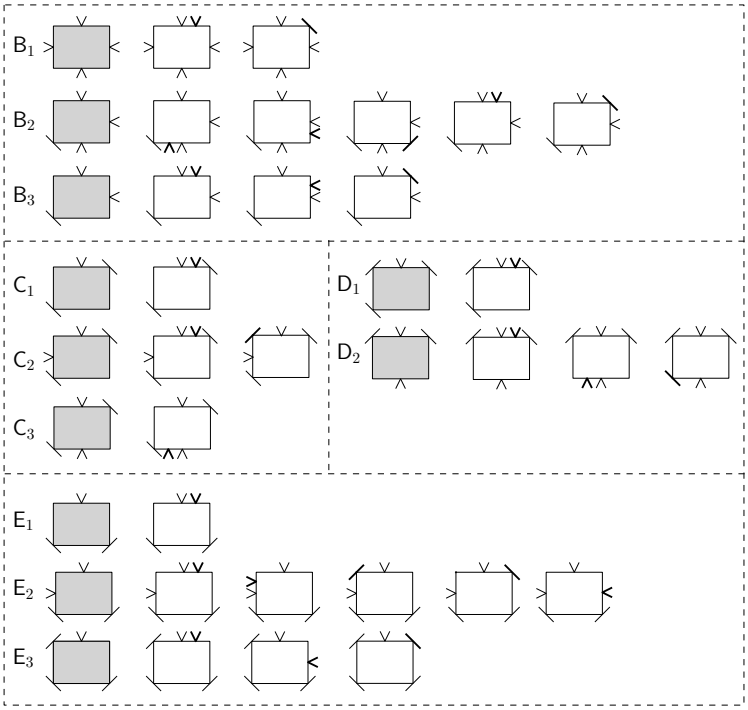

Figure 4 Canonical (sub)types (gray rectangles) and their breaking configurations without duplication. The breaking configurations of subtypes F_1 and F_2 appear as breaking configurations of other types: By adding a sc to a DS Z of type F_1, Z becomes a BC of type either D_1 or E_3. By adding a cc to a DS Z of type F_1, Z becomes a BC of type F_2. By adding a sc to a DS Z of type F_2, Z becomes a BC (of the last type) of type D_1.

References

1. Alok Aggarwal and Joel Martin Wein. Computational Geometry Lecture Notes for MIT, 1988.
2. Helmut Alt, David Hsu, and Jack Snoeyink. Computing the Largest Inscribed Isothetic Rectangle. In *Proceedings of 7th Canadian Conference on Computational Geometry (CCCG 1995)*, pages 67–72. University of British Columbia, 1995.
3. Nina Amenta. Bounded Boxes, Hausdorff Distance, and a New Proof of an Interesting Helly-type Theorem. In *Proceedings of 10th Annual Symposium on Computational Geometry (SoCG 1994)*, pages 340–347, 1994.
4. Sang Won Bae, Chunseok Lee, Hee-Kap Ahn, Sunghee Choi, and Kyung-Yong Chwa. Computing minimum-area rectilinear convex hull and L-shape. *Computational Geometry*, 42(9):903–912, 2009.
5. Ralph P. Boland and Jorge Urrutia. Finding the Largest Axis-Aligned Rectangle in a Polygon in $O(n \log n)$ time. In *Proceedings of 13th Canadian Conference on Computational Geometry (CCCG 2001)*, pages 41–44, 2001.
6. Sergio Cabello, Otfried Cheong, Christian Knauer, and Lena Schlipf. Finding largest rectangles in convex polygons. *Computational Geometry*, 51:67–74, 2016.
7. Danny Z. Chen and Haitao Wang. Visibility and ray shooting queries in polygonal domains. *Computational Geometry*, 48(2):31–41, 2015.
8. Karen Daniels, Victor Milenkovic, and Dan Roth. Finding the largest area axis-parallel rectangle in a polygon. *Computational Geometry*, 7(1):125–148, 1997.
9. Paul Fischer and Klaus-Uwe Höffgen. Computing a maximum axis-aligned rectangle in a convex polygon. *Information Processing Letters*, 51(4):189–193, 1994.
10. Partha P. Goswami, Sandip Das, and Subhas C. Nandy. Triangular range counting query in 2D and its application in finding k nearest neighbors of a line segment. *Computational Geometry*, 29(3):163–175, 2004.
11. Olaf Hall-Holt, Matthew J. Katz, Piyush Kumar, Joseph S. B. Mitchell, and Arik Sityon. Finding Large Sticks and Potatoes in Polygons. In *Proceedings of 17th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA 2016)*, pages 474–483, 2006.
12. Christian Knauer, Lena Schlipf, Jens M. Schmidt, and Hans Raj Tiwary. Largest inscribed rectangles in convex polygons. *Journal of Discrete Algorithms*, 13:78–85, 2012.
13. Michael McKenna, Joseph O'Rourke, and Subhash Suri. Finding the largest rectangle in an orthogonal polygon. In *Proceedings of 23rd Allerton Conference on Communication, Control and Computing*, pages 486–495, 1985.
14. Derick Wood and Chee K. Yap. The orthogonal convex skull problem. *Discrete & Computational Geometry*, 3(4):349–365, 1988.