8. Point location

Introduction. For a given planar subdivision (map), we want to find a search structure that finds in which face (region) a point lies when entering the point coordinates. The idea of the algorithm is to "refine" the subdivision into a so-called *trapezoidal map*, and to construct the search structure for this subdivision. The advantage of the trapezoidal map is that its individual areas are trapezoids and triangles (which can be considered as deformed trapezoids), and these can be relatively simply described using only four data.

FIGURE 8.1 Creation of a trapezoidal map.

Trapezoidal map. Let us have a planar subdivision (usually given by a double-connected edge list) bounded by a rectangle R. This subdivision with its edges determines a set of segments that do not intersect at the inner points, but may have common end points. We construct the trapezoidal map for any such set of segments $S = \{s_1, s_2, \dots s_n\}$ inside a rectangle R. To make things geometrically easier we assume that no two end points of these segments have the same x-coordinate. We will remove this limiting assumption later. The left end of the segment s_i will be p_i, the right end point q_i.

We build the trapeziodal map $\mathcal{T}(S)$ so that through each end point of a segment from S we lead a vertical segment connecting the nearest upper segment with the nearest lower segment, see Figure 8.2. This divides the entire rectagle R into trapezoids and triangles.

FIGURE 8.2 A trapezoidal map for three segments.

Let's show how the trapezoids and triangles of this map can be described using segments from S and their end points. Each trapezoid or triangle Δ has two non-vertical sides, each being a part of a segment from S or a part of the upper or lower side of the rectangle R. The upper side of this trapezoid will be called top(Δ), the lower one will be denoted as bottom(Δ). The vertical sides are determined by end points p_j and q_j of segments from S. In the case of a triangle, one of the vertical side reduces to a point. Therefore, every trapezoid or triangle Δ is determined by a top and a bottom and by a point leftp(Δ) through which the left vertical side passes and by a point rightp(Δ) through which the right vertical side passes.

FIGURE 8.3 Description of the trapezoid Δ.

The substantial is that in the trapezoidal map for n segments the number of vertices and trapezoids is linear in n.

Theorem 8.1. *The trapezoidal map for n segments has at most $6n + 4$ vertices and $3n + 1$ trapezoids.*

Proof. The overall number of vertices is given by the vertices of the rectagle R – they are 4, by end points of segments – they are at most $2n$ and by new vertices, which were made by at most $2n$ vertical segments – they are twice the number of verticals, so at most $4n$. Altogether we get at most $6n + 4$ points.

We find the number of trapezoids by specifying for each point the maximum number of trapezoids Δ for which this point is the left point leftp(Δ). There is just one such trapezoid for the left lower corner of the rectangle R. There is one such trapezoid for a right end point q_i, there are two such trapezoids for a left end point p_i.

FIGURE 8.4 Numbers of trapezoids for given left point leftp.

Taking into account that some of end points coincide (in this case some trapezoids are counted twice), the number of all trapezoids will be at most

$$1 + n + 2n = 3n + 1.$$

□

Search structure. We are given a point r which does not lie on any segment of S and the x-coordinate of which differs from all x-coordinates of the end points p_i and q_i. The search structure for a trapezoidal map $\mathcal{T}(S)$ is used to determine the trapezoid in which the specified point r is located. It is an oriented graph $\mathcal{D}(S)$ with the following properties:

- In leaves of the graph there are all trapezoids of the trapezoidal map $\mathcal{T}(S)$.
- Two edges come from every inner node.
- Inner nodes are of two types.
- The nodes of the first type are connected with end points p_i and q_i. From such a node (denote it for a moment as p) we go left if for x-coordinates $r_x < p_x$, and we go right if $r_x > p_x$.
- The nodes of the second type are described by segments s_i. From such a node we proceed left if the point r lies under the segment s_i, and we go right if r lies over s_i.

While for the specified set S the trapezoid map is uniquely identified, the search structure with the above properties is not uniquely determined. The following figures give simple examples of trapezoidal maps and related search structures.

FIGURE 8.5 Trapezoidal map and a serch structure for 1 segment.

FIGURE 8.6 Trapezoidal map and a serch structure for 2 segment.

The length of a path from the root to a leaf corresponds to the time for finding that the specified point is lying in the trapezoid determined by this leaf. Notice the substantially different lengths of paths to various leaves.

Randomized incremental algorithm. The trapezoidal map for an empty set of segments consists of only the rectangle R, the search structure contains only a single node which is a leaf corresponding to the rectangle R. Denote

$$S_i = \{s_1, s_2, \ldots, s_i\}.$$

The idea of the algorithm is to create a trapezoidal map $\mathcal{T}(S_i)$ and a search structure $\mathcal{D}(S_i)$ for the set of segments S_i from the trapezoidal map and the search structure for the set S_{i-1}. Because the search structure depends on the order of the segments, we take this order randomly. The algorithm is framed in the following pseudocode:

ALGORITHM. TrapezoidalMap from pseudo.pdf, page 28.

Following segment. We now show how the algorithm briefly described in the 4th line of the previous pseudocode works. We have a trapezoidal map $\mathcal{T}$ and a search structure $\mathcal{D}$ for the set S_{i-1}. Into the rectangle R we add the segment $s = s_i$ with the left end point p and the right end point q. We want to find all trapezoids Δ_0, Δ_1, $\dots$, Δ_k which the segment s intersects from left to right

FIGURE 8.7 The segment s passes through the trapezoids $\Delta_0, \Delta_1, \Delta_2, \Delta_3, \Delta_4$.

To describe this algorithm we will need the notion of adjacent trapezoids and left and right neighbours. We say that two trapezoids are adjacent if they have a common part of the vertical side that does not degenerate to a point.

FIGURE 8.8 Two couples of trapezoids Δ_0 and Δ_1, Δ_0 and Δ_3 are adjacent, while the couples Δ_0 and Δ_2 or Δ_1 and Δ_2 are not adjacent.

If two adjacent trapezoids have a common bottom, we say that one is the *lower* left or right *neighbour* of the other. If they have a common top, we say that one is the *upper* left or the right *neighbour* of the other.

FIGURE 8.9 The trapezoid Δ_1 is the upper right neighbour of the trapezoid Δ_0. The trapezoid Δ_0 is the lower left neighbour of the trapezoid Δ_2.

First the algorithm finds a trapezoid Δ_0 in which the left end point p of the segment s lies. If the point p is not the left end point of any segment from S_{i-1}, simply use the search structure $\mathcal{D}(S_{i-1})$ to find Δ_0. If p is the left end point of one or more segments of S_{i-1}, there are more trapezoids to the right of it, and we have to select that one through which the segment s runs. This is done by comparing the slopes of these segments with the slope of s. The slope of this segment with the end points p and q is equal to the quotient of the differences of y- and x-coordinates

$$\frac{q_y - p_y}{q_x - p_x}.$$

The traced trapezoid Δ_0 is characterized by the fact that its top side slope is bigger and its bottom side slope is smaller than the slope of the segment s.

FIGURE 8.10 The slope of the segment s_1 is bigger than the slope of s and this is bigger than the slope of s_2. The requested trapezoid Δ_0 is the trapezoid B.

If the end point q lies to the left of rightp(Δ_0), the whole segment s is contained in Δ_0.

FIGURE 8.11 The whole segment s lies in Δ_0.

If the end point q lies to the right of rightp(Δ_0), the segment s intersects another trapezoid Δ_1. This trapezoid is the lower right neighbour of Δ_0, if rightp(Δ_0) lies over s, and the upper right neighbour, if rightp(Δ_0) lies under s. See the next picture.

FIGURE 8.12 Specifying Δ_1 according to relative position of the segment s and the point rightp(Δ_0).

In the same way we proceed further:

ALGORITHM FollowSegment from pseudo.pdf, page 29.

Updating the trapezoidal map and the search structure. In this section, we show the basic idea of the algorithm that implements the 5th and 6th lines of the TrapezoidalMap algorithm, i.e. how to replace trapezoids Δ_0, Δ_1, $\ldots, \Delta_k$ intersected by the segment $s = s_i$ by new trapezoids and so to change the trapezoidal map $\mathcal{T}$ and how to change related search structure $\mathcal{D}$.

Suppose first that the whole segment s lies inside one trapezoid Δ_0.

FIGURE 8.13 The whole segment s lies inside the trapezoid Δ_0.

In this case the trapezoid Δ_0 in $\mathcal{T}$ is replaced by the trapezoids L, H, D a P. Each of them is determined by its top and bottom and by its left and right points. For example, for the trapezoid H we have

$$\mathrm{top}(H) = \mathrm{top}(\Delta_0),\ \mathrm{bottom}(H) = s,\ \mathrm{leftp}(H) = p,\ \mathrm{rightp}(H) = q.$$

We carry out the modification of the related search structure in the way shown by the following figure.

FIGURE 8.14 The transition from $\mathcal{D}(S_{i-1})$ to $\mathcal{D}(S_i)$.

Now let the segment s intersects more trapezoids. For the sake of clarity, consider the situation illustrated in the first of the two following figures.

FIGURE 8.15 Transition from $\mathcal{T}(S_{i-1})$ to $\mathcal{T}(S_i)$.

We replace the trapezoids Δ_0, Δ_1, Δ_2, Δ_3 by new trapezoids as follows. L is the trapezoid lying to the left of the point p. It has the same top, bottom a leftp as Δ_0. Since $\mathrm{rightp}(\Delta_0)$ lies under s, the next trapezoid is D^1 with

$$\mathrm{leftp}(D^1) = p,\ \mathrm{rightp}(D^1) = \mathrm{rightp}(\Delta_0),\ \mathrm{top}(D^1) = s,\ \mathrm{bottom}(D^1) = \mathrm{bottom}(\Delta_0).$$

The right point of Δ_1 lies over s, hence the next trapezoid lies over s. Denote it H^1. It is determined by the data

$$\mathrm{leftp}(H^1) = p,\ \mathrm{bottom}(H^1) = s,\ \mathrm{top}(H^1) = \mathrm{top}(\Delta_1),\ \mathrm{rightp}(H^1) = \mathrm{rightp}(\Delta_1).$$

The right point of Δ_2 lies again over s, so the next new trapezoid H^2 will lie over s and

$$\mathrm{leftp}(H^2) = \mathrm{rightp}(\Delta_1),\ \mathrm{bottom}(H^2) = s,\ \mathrm{top}(H^2) = \mathrm{top}(\Delta_2), \mathrm{rightp}(H^2) = \mathrm{rightp}(\Delta_2).$$

In the further trapezoid Δ_3 there is the right end point q of the segment s. We complete the list of new trapezoids by the trapezoid D^2 lying under s with the right point equal to q, by the trapezoid H^3 lying over s with the right point q and by the trapezoid P lying to the right of q.

The modification of the search structure is illustrated by the following figure.

FIGURE 8.16 The transition from $\mathcal{D}(S_{i-1})$ to $\mathcal{D}(S_i)$.

We will not give explicit pseudocodes for updating the trapezoidal map $\mathcal{T}$ and the related search structure $\mathcal{D}$. The reader can create them on the basis of the previous considerations or he/she can find them in the bachelor thesis (in Czech) by Ondřej Folvarčný

http://is.muni.cz/auth/th/211164/prif_b/BP.pdf?lang=cs

on pages 20 and 21.

How to remove restrictive assumptions. The simplification consisted in the fact that the different end points of the segments and a considered point r had different x-coordinates. We will remove this assumption by introducing shear transformation

$$\varphi(x, y) = (x + \varepsilon y, y)$$

for small $\varepsilon > 0$.

Lemma 8.2. *For a finite set P of points there is an $\varepsilon > 0$ such that for any two points $p, q \in P$ it holds*

(1) $\varphi(p)_x \neq \varphi(q)_x$,
(2) $p_x < q_x \ \Rightarrow\ \varphi(p)_x < \varphi(q)_x$.

Proof. Let $p_x < q_x$. If $p_y \leq q_y$, then the enequality $\varphi(p)_x < \varphi(q)_x$ holds for all $\varepsilon > 0$.

If $p_y > q_y$, then the enequality $\varphi(p)_x < \varphi(q)_x$ holds if and only if

$$0 < \varepsilon < \frac{q_x - p_x}{p_y - q_y}.$$

Since we choose the points p, q from a finite fixed set, we can find sufficiently small positive ε satisfying the assertion of Lemma.

If now two different points p a q satisfy $p_x = q_x$, it has to be $p_y \neq q_y$, and consequently also $\varphi(p)_x \neq \varphi(q)_x$. □

Now, in the previous considerations, we could replace the arrangement by x-coordinate with the arrangement by x-coordinate of φ. In fact, there is no need to compute an ε to determine the transformation φ.

Lemma 8.3. *For each finite set of points P and a sufficiently small $\varepsilon > 0$, the arrangement of points $p \in P$ by x-coordinates of $\varphi(p)$ is the same as the lexicographic arrangement of points p first according to the x-coordinate and then by the y-coordinate.*

Proof. Let $p < q$ in the given lexicographic order. Then either $p_x < q_x$ and then according to the previous considerations $\varphi(p)_x < \varphi(q)_x$, or $p_x = q_x$ and $p_y < q_y$ and so $\varphi(p)_x < \varphi(q)_x$ as well. □

From the previous considerations, we can make this practical conclusion: *Whenever in previous considerations we find out whether a given point is to the left or to the right of another point, we will use the described lexicographic arrangement. Whenever we find out whether a point lies over or under a segment, we use the arrangement with respect to the y-coordinate.*

Complexity estimates. Since our algorithm is random, we estimate the expected complexity both of the construction of the search structure and of the memory needed to store the search structure. Moreover, we estimate the expected running time for searching a position of a point in the trapezoidal map. Let's start with the last item.

Theorem 8.4. *The expected running time for searching in the search structure $\mathcal{D}$ created for a random order of n segments is $O(\log n)$.*

Proof. The search structure $\mathcal{D}$ is created in n steps. Consider a path for searching the position of a point r. Let X_i is the number of nodes in this path which were added to the search structure in the i-th step. From figures 8.14 and 8.16 it is seen that

$$0 \leq X_i \leq 3.$$

Consider X_i as a random variable. Then the expected time for searching is

$$E\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} E(X_i) \leq 3 \text{ probability } (X_i \neq 0).$$

The fact that $X_i \neq 0$ means, that the point r lies in $\mathcal{T}(S_i)$ in some trapezoid Δ which arises just in the i-th step when adding the segment s_i. At least one of the following options happens:

$$\text{top}(\Delta) = s_i,\ \text{bottom}(\Delta) = s_i,\ \text{leftp}(\Delta) = p_i \text{ or } q_i,\ \text{rightp}(\Delta) = p_i \text{ or } q_i.$$

So the probability that $X_i \neq 0$ is at most $4/i$. The expected time for searching can be estimated from above by the sum

$$3\left(\sum_{i=1}^{n} \frac{4}{i}\right) = 12\left(1 + \sum_{i=2}^{n} \frac{1}{i}\right) \leq 12\left(1 + \int_1^n \frac{dx}{x}\right) = 12(1 + \log n) = O(\log n).$$

The middle estimate using integral follows from the fact that the sum of areas of rectangles with sides 1 and $1/i$ pro $i = 2, 3, \ldots, n$ is smaller then the area of the region between the axis x and the graph of the function $1/x$ between numbers 1 and n on the axis x. See the figure:

FIGURE 8.17 The comparison of the sum $\frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$ with the intergral $\int_1^n \frac{dx}{x}$. □

Theorem 8.5. *The expected size of the search structure is $O(n)$.*

Proof. The size of the search structure is

$$\text{number of leaves } + \sum_{i=1}^{n} \text{ number of inner nodes created in the step } i.$$

Let u_i be the number of trapezoids which arise after adding the segment s_i in the i-th step. In Figures 8.13, 14, 15, 16, you can see that the number of newly created internal nodes in the i-th step is $u_i - 1$. It can be rigorously prooved by induction as it is done on the page 29 of the bachalor thesis by O. Folvarčný. Therefore, assuming u_i to be a random variable, we can estimate the expected size of the data structure from above by the expression

$$O(n) + \sum_{i=1}^{n} E(u_i).$$

To complete the proof it suffices to prove that $E(u_i) = O(1)$.

Let $\Delta \in \mathcal{T}(S_i)$ be a trapezoid and $s \in S_i$ one of the segments. Put

$$\lambda(\Delta, s) = \begin{cases} 1, & \text{if } \Delta \text{ arises by adding the segment } s, \\ 0, & \text{in opposite case.} \end{cases}$$

Since the trapezoid Δ is determined at most by 4 segments from S, we get

$$\sum_{s \in S_i} \lambda(\Delta, s) \leq 4.$$

That is why

$$\sum_{s \in S_i} \sum_{\Delta \in \mathcal{T}(S_i)} \lambda(\Delta, s) \leq 4|\mathcal{T}(S_i)| \leq 4(3i+1) = O(i).$$

Here $|\mathcal{T}(S_i)|$ is the number of trapezoids in $\mathcal{T}(S_i)$ which we estimated by the number $3i+1$ in Theorem 8.1. Now realize that the number

$$\sum_{\Delta \in \mathcal{T}(S_i)} \lambda(\Delta, s_i)$$

gives the number of trapezoids which arise in the i-th step. Hence

$$E(u_i) = \frac{1}{i} \left(\sum_{s \in S_i} \Big(\sum_{\Delta \in \mathcal{T}(S_i)} \lambda(\Delta, s) \Big) \right) = \frac{O(i)}{i} = O(1).$$

□

Theorem 8.6. *The algorithm constructs the search structure in expected time $O(n \log n)$.*

Proof. The time for creating $\mathcal{T}(S_i)$ and $\mathcal{D}(S_i)$ from $\mathcal{T}(S_{i-1})$ and $\mathcal{D}(S_{i-1})$ is $O(u_i)$ plus a time for searching in which trapezoid from $\mathcal{T}(S_{i-1})$ the left end point p_i of the segment s_i is lying. Its expected value we can estimate using the previous computations in this way:

$$\sum_{i=1}^{n} \big(O(E(u_i)) + O(\log i) \big) = O(n) + O\Big(\sum_{i=1}^{n} \log i \Big) \leq O(n) + O(n \log n) = O(n \log n).$$

□

Conclusion. The original task was to find in which face of a fixed plane subdivision a given point lies. So far we have found in which trapozoid of the trapezoidal map the point lies. Now it is easy to find a face in which the found trapezoid is contained. It can be done if we find in which cycle of the corresponding edge-connected list the bottom of the trapezoid (taken with suitable orientation) lies.