Lecture 2b: Arrangements of lines

David Eppstein
University of California, Irvine

Spring Quarter, 2022
Minimum-area triangles
The Heilbronn triangle problem

Find $n$ points in unit square maximizing the minimum area triangle

3 in line $\Rightarrow$ area $= 0$, so we want as far out of line as possible

Example: $n = 6$, minimum area $= 1/8$
Known bounds on the Heilbronn problem

Sort points by $x$-coordinates $\Rightarrow$ some three points lie in a thin vertical strip $\Rightarrow$ their triangle has area $O(1/n)$

For some $c > 0$, every point set has a triangle with area

$$\Delta(n) \leq \frac{\exp \left( c \sqrt{\log n} \right)}{n^{8/7}}$$

[Komlós et al. 1981]

Some $n$-point sets have smallest triangle area

$$\Omega \left( \frac{\log n}{n^2} \right)$$

[Komlós et al. 1982]

Unsolved problem: close this big gap!
How to tell how good a solution is?

Algorithmic problem:
Given \( n \) points,
find their minimum-area triangle

Naïve algorithm:
Use the triangle-area primitive on all triples
Time = \( O(n^3) \)

We can do much better!
From triangle area to vertical distance

For each two (non-vertically-aligned) points $p$, and $q$:

▶ We can find the line $\ell : y = mx + b$ through $p$ and $q$

▶ Area of $pqr$ is $\frac{1}{2}$\text{width} \cdot \text{height}$ where width is $|x_p - x_q|$ and height (vertical distance of $r$ from $\ell$) is $|y_r - (mx_r + b)|$, the difference between the actual $y$-coordinate of $r$ and the $y$-coordinate predicted by the line given $x_r$

▶ Do this for all $r \Rightarrow$ find the smallest triangle using $p$ and $q$

▶ But this is still $O(n^3)$ because there are $O(n^2)$ lines to try
Duality of vertical distance

Given points with coordinates \((x, y)\)
and lines with coordinates \((m, b)\)
⇒ (signed) vertical distance is \(y - mx - b\)

Given points with coordinates \((m, -b)\)
and lines with coordinates \((x, -y)\)
⇒ (signed) vertical distance is \(-b - mx + y\)

Using point coordinates for lines and vice versa
(and negating the second coordinate)
doesn’t change our calculation!
Original and dual problems

**Original problem:**

Given \( n \) points, no two \( x \)-coordinates equal
Find the line through each pair
Find third point with minimum vertical distance to it
(Min-area triangle will be one of the triples we find)

**Equivalent dual problem:**

Given \( n \) lines, no two parallel
Find the crossing point of each two lines
Find third line with minimum vertical distance to it
(Min-area triangle will still be one of the triples we find)
Use the plane sweep algorithm we already know about for finding crossings of line segments, on the $n$ given lines.

Each time it sweeps over a crossing, use binary search tree (of lines meeting vertical sweep line) to find the two lines above and below the crossing; one of the two will have minimum vertical distance.

$O(n^2)$ crossings $\Rightarrow$ $O(n^2)$ priority queue and binary search tree operations $\Rightarrow$ total time $O(n^2 \log n)$

We can do better!
Arrangements of lines
An arrangement of lines

With no two parallel lines, every two lines cross

Convenient to include one more line at infinity, proj. coords \((0, 0, 1)\)
Incremental construction algorithm

- Start with DCEL describing only the line at infinity
- For each line $\ell_i$ in arbitrary order:
  - Walk along the current subdivision of the line at infinity to find an infinite cell where the new line starts
  - For each cell crossed by the new line, walk around the cell to find where it exits the cell, and use that information to split the cell into two smaller cells

Claim: This algorithm constructs line arrangements in $O(n^2)$ time

Easy to modify to find the lines above/below each crossing point
⇒ we can use it to find minimum-area triangles in $O(n^2)$ time

How to prove its running time?
By understanding how many things are in the arrangement we can analyze algorithms that do something for each thing

\[ \# \text{ vertices (crossings)} \leq \binom{n}{2} = O(n^2) \]

Each line divided into 2 rays and \( \leq n - 2 \) segments \( \Rightarrow \)
\[ \# \text{ rays} = 2n \]
\[ \# \text{ segments} \leq n(n - 2) = O(n^2) \]

Each cell has a bottom vertex or extends infinitely far down
Infinite-downward cells are separated by \( n \) lines \( \Rightarrow \)
\[ \# \text{ cells} \leq \binom{n}{2} + n + 1 = O(n^2) \]
Zones

The zone of a line is the set of all cells that it touches.

Its complexity (sum of cell complexities) controls the time taken for walks across cells when adding the line to the arrangement.

Zone theorem: All zones have complexity $O(n)$.
Ideas for proof of zone theorem

Rotate so the given line is horizontal
Add other lines to zone, sorted by slope (negative to positive)

Each line creates $\leq 2$ edges on left sides of lines, above given line:
One on new line, one formed by splitting an existing edge

Repeat same argument for right sides of lines and sides below the given line, and separately count zone edges on the given line itself
Analysis of incremental arrangement construction

Each zone has complexity $O(n)$

Adding each line to the arrangement takes time proportional to its zone (in the arrangement so far), $O(n)$

Constructing the whole arrangement takes time $O(n^2)$

Finding the minimum-area triangle takes time $O(n^2)$
The middle level problem

Middle level = cells with $\lfloor n/2 \rfloor$ lines above and $\lceil n/2 \rceil$ lines below

Big unsolved problem: What is its complexity?

$O(n^{4/3})$ [Dey 1998] $\Omega \left(n e^{c(\log k)^{1/2}}\right)$ [Tóth 2001]

