Computational Geometry

Lecture 4: Smallest enclosing circles and more
Facility location

Given a set of houses and farms in an isolated area. Can we place a helicopter ambulance post so that each house and farm can be reached within 15 minutes?

Where should we place an antenna so that a number of locations have maximum reception?
Facility location in geometric terms

Given a set of points in the plane. Is there any point that is within a certain distance of these points?

Where do we place a point that minimizes the maximum distance to a set of points?
Facility location in geometric terms

Given a set of points in the plane, compute the smallest enclosing circle
Observation: It must pass through some points, or else it cannot be smallest

- Take any circle that encloses the points, and reduce its radius until it contains a point $p$
- Move center towards $p$ while reducing the radius further, until the circle contains another point $q$
Smallest enclosing circle

- Move center on the bisector of $p$ and $q$ towards their midpoint, until:
  (i) the circle contains a third point, or
  (ii) the center reaches the midpoint of $p$ and $q$
Question: Does the “algorithm” of the previous slide work?
Smallest enclosing circle

**Observe:** A smallest enclosing circle has (at least) three points on its boundary, or only two in which case they are diametrically opposite.

**Question:** What is the extra property when there are three points on the boundary?
Randomized incremental construction

Construction by randomized incremental construction

*Incremental construction*: Add points one by one and maintain the solution so far

*Randomized*: Use a random order to add the points
Putting in random order

The points may be given in any order, the algorithm will just reorder them

- Let $j$ be a random integer in $[1, n]$
- Swap $p_j$ and $p_n$
- Recursively shuffle $p_1, \ldots, p_{n-1}$

Putting in random order takes $O(n)$ time
Expected running time

Every one of the $n!$ orders is equally likely

The expected time taken by the algorithm is the average time over all orders

$$\frac{1}{n!} \cdot \sum_{\Pi \text{ permutation}} \text{time if the random order is } \Pi$$
Adding a point

Let $p_1, \ldots, p_n$ be the points in random order.

Let $C_i$ be the smallest enclosing circle for $p_1, \ldots, p_i$.

Suppose we know $C_{i-1}$ and we want to add $p_i$.

- If $p_i$ is inside $C_{i-1}$, then $C_i = C_{i-1}$
- If $p_i$ is outside $C_{i-1}$, then $C_i$ will have $p_i$ on its boundary.
Adding a point

Adding a point to the smallest enclosing circle algorithm.
Adding a point

**Question:** Suppose we remembered not only $C_{i-1}$, but also the two or three points defining it. It looks like if $p_i$ is outside $C_{i-1}$, the new circle $C_i$ is defined by $p_i$ and some points that defined $C_{i-1}$. Why is this false?
Adding a point
Adding a point

How do we find the smallest enclosing circle of $p_1, \ldots, p_{i-1}$ with $p_i$ on the boundary?

We study the *new(!) geometric problem of computing the smallest enclosing circle with a given point $p$ on its boundary.*
Given a set $P$ of points and one special point $p$, determine the smallest enclosing circle of $P$ that must have $p$ on the boundary.

**Question:** How do we solve it?
Construction by *randomized incremental construction*

**incremental construction**: Add points one by one and maintain the solution so far

**randomized**: Use a random order to add the points
Adding a point

Let $p_1, \ldots, p_{i-1}$ be the points in random order

Let $C'_j$ be the smallest enclosing circle for $p_1, \ldots, p_j$ ($j \leq i - 1$) and with $p$ on the boundary

Suppose we know $C'_{j-1}$ and we want to add $p_j$

- If $p_j$ is inside $C'_{j-1}$, then $C'_j = C'_{j-1}$
- If $p_j$ is outside $C'_{j-1}$, then $C'_j$ will have $p_j$ on its boundary (and also $p$ of course!)
Adding a point

Computational Geometry  Lecture 4: Smallest enclosing circles and more
Adding a point

How do we find the smallest enclosing circle of \( p_1 \ldots , p_{j-1} \) with \( p \) and \( p_j \) on the boundary?

We study the \textit{new(!)} geometric problem of computing the smallest enclosing circle with two given points on its boundary.
Given a set $P$ of points and two special points $p$ and $q$, determine the smallest enclosing circle of $P$ that must have $p$ and $q$ on the boundary.

**Question:** How do we solve it?
Two points known
Two points known
Algorithm: two points known

Assume w.l.o.g. that $p$ and $q$ lie on a vertical line. Let $\ell$ be the line through $p$ and $q$ and let $\ell'$ be their bisector.

For all points left of $\ell$, find the one that, together with $p$ and $q$, defines a circle whose center is leftmost $\rightarrow p_l$.

For all points right of $\ell$, find the one that, together with $p$ and $q$, defines a circle whose center is rightmost $\rightarrow p_r$.

Decide if $C(p, q, p_l)$ or $C(p, q, p_r)$ or $C(p, q)$ is the smallest enclosing circle.
Two points known

\[ C(p, q, p_r) \]

\[ C(p, q, p_l) \]
Analysis: two points known

Smallest enclosing circle for $n$ points with two points already known takes $O(n)$ time, worst case.
Algorithm: one point known

- Use a random order for $p_1, \ldots, p_n$; start with $C_1 = C(p, p_1)$
- for $j \leftarrow 2$ to $n$ do
  - If $p_j$ in or on $C_{j-1}$ then $C_j = C_{j-1}$; otherwise, solve smallest enclosing circle for $p_1, \ldots, p_{j-1}$ with two points known ($p$ and $p_j$)
Analysis: one point known

If only one point is known, we used randomized incremental construction, so we need an *expected time analysis*.
**Analysis: one point known**

**Backwards analysis:** Consider the situation *after* adding $p_j$, so we have computed $C_j$.
Analysis: one point known

The probability that the $j$-th addition was expensive is the same as the probability that the smallest enclosing circle changes (decreases in size) if we remove a random point from the $j$ points.
Analysis: one point known

This probability is $2/j$ in the left situation and $1/j$ in the right situation.
Analysis: one point known

The expected time for the $j$-th addition of a point is

$$\frac{j - 2}{j} \cdot \Theta(1) + \frac{2}{j} \cdot \Theta(j) = O(1)$$

or

$$\frac{j - 1}{j} \cdot \Theta(1) + \frac{1}{j} \cdot \Theta(j) = O(1)$$

The expected running time of the algorithm for $n$ points is:

$$\Theta(n) + \sum_{j=2}^{n} \Theta(1) = \Theta(n)$$
Analysis: one point known

Smallest enclosing circle for $n$ points with one point already known takes $\Theta(n)$ time, expected.
Algorithm: smallest enclosing circle

- Use a random order for \( p_1, \ldots, p_n \); start with \( C_2 = C(p_1, p_2) \)
- for \( i \leftarrow 3 \) to \( n \) do
  - If \( p_i \) in or on \( C_{i-1} \) then \( C_i = C_{i-1} \); otherwise, solve smallest enclosing circle for \( p_1, \ldots, p_{i-1} \) with one point known (\( p_i \))

\[ C_{i-1} \]
\[ p_i \]
\[ C_i \]
For smallest enclosing circle, we used randomized incremental construction, so we need an *expected time analysis*. 
**Backwards analysis:** Consider the situation *after* adding $p_i$, so we have computed $C_i$
The probability that the $i$-th addition was expensive is the same as the probability that the smallest enclosing circle changes (decreases in size) if we remove a random point from the $i$ points.
Analysis: smallest enclosing circle

This probability is $3/i$ in the left situation and $2/i$ in the right situation.
Analysis: smallest enclosing circle

The expected time for the $i$-th addition of a point is

$$\frac{i - 3}{i} \cdot \Theta(1) + \frac{3}{i} \cdot \Theta(i) = O(1)$$

or

$$\frac{i - 2}{i} \cdot \Theta(1) + \frac{2}{i} \cdot \Theta(i) = O(1)$$

The expected running time of the algorithm for $n$ points is:

$$\Theta(n) + \sum_{i=3}^{n} \Theta(1) = \Theta(n)$$
Theorem The smallest enclosing circle for \( n \) points in plane can be computed in \( O(n) \) expected time.
When does it work?

Randomized incremental construction algorithms of this sort (compute an ‘optimal’ thing) work if:

1.) The test whether the next input object violates the current optimum must be possible and fast

2.) If the next input object violates the current optimum, finding the new optimum must be an easier problem than the general problem

3.) The thing must already be defined by $O(1)$ of the input objects

4.) Ultimately: the analysis must work out
Diameter: Given a set of $n$ points in the plane, compute the two points furthest apart

Closest pair: Given a set of $n$ points in the plane, compute the two points closest together
**Width:** Given a set of $n$ points in the plane, compute the smallest distance between two parallel lines that contain the points (narrowest strip)
Rotating callipers

The width can be computed using the rotating callipers algorithm

- Compute the convex hull
- Find the highest and lowest point on it; they define two horizontal lines that enclose the points
- Rotate the lines together while proceeding along the convex hull
Rotating callipers

![Diagram of rotating callipers](image)
Rotating callipers
Rotating callipers
Rotating callipers
Rotating callipers
Rotating callipers
Property: The width is always determined by three points of the set.

Theorem: The rotating callipers algorithm determines the width (and the diameter) in $O(n \log n)$ time.
**Property:** The width is always determined by three points of the set

We can maintain the two lines defining the width to have a fast test for violation
Adding a point

**Question:** How about adding a point? If the new point lies inside the narrowest strip we are fine, but what if it lies outside?
Adding a point
Adding a point
A good reason to be very suspicious of randomized incremental construction as a working approach is *non-uniqueness* of a solution.
Minimum bounding box

**Question:** Can we compute the minimum axis-parallel bounding box by randomized incremental construction?
Yes, in $O(n)$ expected time

... but a normal incremental algorithm does it in $O(n)$ worst case time
Problem 1: Given $n$ disks in the plane, can we compute the lowest point in their common intersection efficiently by randomized incremental construction?

Problem 2: Given $n$ disks in the plane, can we compute the lowest point in their union efficiently by randomized incremental construction?
Problem: Given a set of $n$ half-planes, can we decide efficiently if their intersection is empty?
**Problem:** Given a set of $n$ red and blue points in the plane, can we decide efficiently if they have a separating line?