Recall

Voronoi diagram

Given points $P = \{p_1, \ldots, p_n\}$ in the plane, the **Voronoi region** of $p_i$ is

$$V(p_i) = \{x \in \mathbb{R}^2 : d(x, p_i) \leq d(x, p_j) \forall j \neq i\}$$

$p_i$ is called a **site**.

The **Voronoi diagram** $\mathcal{V}(P)$ consists of all the Voronoi regions.

Given points $P = \{p_1, \ldots, p_n\}$ in the plane, the **Delaunay triangulation** $\mathcal{D}(P)$ is a graph with vertices $p_1, \ldots, p_n$ and edge $(p_i, p_j)$ iff $V(p_i)$ and $V(p_j)$ share an edge.

$\mathcal{D}(P)$ is the **planar dual** of $\mathcal{V}(P)$.
Recall Delaunay triangulation and empty circle property: \((p,q)\) is an edge of the Delaunay triangulation iff there is an empty circle through \(p\) and \(q\).

An algorithmically more useful characterization:

**Lemma.** A triangulation is Delaunay iff every edge \(e=(p,q)\) is legal.

**Definition.** edge \(e=(p,q)\) is *legal* if either:
- \(e\) is on the convex hull or
- \(e\) is interior with triangles \(pqr\) and \(pqs\), and \(r\) is not in \(\text{Circle}(pqs)\)

**Note:** \(r\) in \(\text{Circle}(p,q,s)\) iff \(s\) in \(\text{Circle}(p,q,r)\)

Note that this is a condition about ALL edges, not a single edge:
edge \(e\) is Delaunay (\(\exists\) an empty circle through its endpoints) \(\Rightarrow\) \(e\) is legal \(\Leftarrow\)
Lemma. A triangulation is Delaunay iff every edge $e=(p,q)$ is legal.

Proof.

$\Rightarrow$ prove $e$ illegal $\Rightarrow e$ not Delaunay  
(no empty circle)

$\Rightarrow$ every circle through $pq$ contains $r$ or $s$.

So $pq$ not a Delaunay edge.

$\Leftarrow$ Suppose triangulation $T$ not Delaunay. Find an illegal edge.

$\exists \triangle pqs$ and site $x$ inside circle$(p,q,s)$

Pick $\triangle pqs$ and site $x$ to minimize distance $x$ to $\triangle pqs$

Consider the triangle $pqr$ on the other side of $pq$  
(exists since $pq$ not on convex hull)

If $r$ inside circle$(p,q,s)$ then $pq$ is illegal.

Consider $\triangle pqr$ and site $x$. $x$ is inside circle and distance $x$ to $\triangle pqr$ is smaller.

Contra. to how we chose the $\Delta$.

$\therefore pq$ is illegal.
What to do with an illegal edge \((p,q)\)

**Edge Flip**

Claim: \((r,s)\) is a legal edge.

\[\text{change Circle (psq) to Circle (psr)}\]
\[\text{It shrinks away from } q\]
\[\text{So } q \text{ outside Circle (psr)}\]
\[\text{So } rs \text{ is legal.}\]

EX: If \(pq\) was the only illegal edge, do we get Delaunay triangulation?
Flipping illegal edges makes global improvements in a triangulation: the **Angle Vector**.

For any triangulation $T$ of a set of points, the *angle vector* $A(T)$ is the list of angles of the triangles sorted min to max.

**example**

![Diagram](image)

$$A(T) = (45, 45, 60, 60, 60, 90)$$

The angle vector always has length $3t$ where $t$ is the number of triangles.

We compare two angle vectors *lexicographically* (dictionary ordering)

**example**

![Diagram](image)

$$A(T) = (45, 45, 60, 60, 60, 90)$$

$$A(T') = (45, 45, 50, 50, 80, 90)$$

so $A(T) > A(T')$
**Lemma.** Flipping an illegal edge increases the angle vector lexicographically.

Thus, flipping illegal edges does not cycle and we eventually get the Delaunay triangulation.

We need:

**Thales Theorem.** For pq a chord of a circle, angle psq is constant for s on an arc of the circle. For s inside, the angle is bigger. For s outside the angle is smaller.

Actually, Thales considered pq to be a diameter. The generalization is in Euclid.

**Lemma.** Flipping an illegal edge increases the angle vector lexicographically.

**Proof.**

Only angles shown here change.

Old angles: \( a, d, c+b > e, h, f+g \)

New angles: \( c, f, d+e, b, g, a+h \) (after flip)

Some comparisons:
- \( c > h \) - Thales on chord \( ps \)
- \( b > e \) - chord \( qs \)
- \( f > a, g > d \) - chord \( qr \)

\( \text{Chord } pr \)

Want: smallest new angle \( \geq \) smallest old angle.

Every new angle is larger than some old angle.

\( c > h \quad d+e > d \quad a+h > a \)

\( f > a \quad b > e \quad g > d \)

\( \therefore \text{ new min } \geq \text{ old min } \)
Thus, flipping illegal edges always gets you to the Delaunay triangulation, and the Delaunay triangulation has the lexicographically maximum angle vector.

Consequences:

**Theorem.** The Delaunay triangulation maximizes the minimum angle.

**Algorithm** to find the Delaunay triangulation: find ANY triangulation and then flip illegal edges until there are none left.

How many flips does this take?

- We will prove that no edge reappears.
- There are $O(n^2)$ edges.
- $\therefore$ at most $O(n^2)$ flips.
**Lemma.** No edge reappears when we flip illegal edges.

**Proof.** Recall connection between Delaunay triangulation and convex hull.

![Diagram of Delaunay triangulation]

Flip illegal edge (a,c) to (d,b).

$d$ inside Circle(abc) \(\Rightarrow\)

Want to show $ac$ never reappears as we flip.

\(\hat{a}\) \(\hat{b}\) \(\hat{c}\) \(\hat{\Delta}\) form a tetrahedron.

Initial triangulation \(\rightarrow\) $\hat{a}\hat{c}\hat{b}$ \(\hat{a}\hat{d}\hat{c}\) are on top of tetrahedron.

Final triangulation \(\rightarrow\) $\hat{d}\hat{b}\hat{c}$, $\hat{d}\hat{a}\hat{b}$ are on bottom of tetrahedron.

So segment $\hat{a}\hat{c}$ disappears above (in 3D) forever.
[Randomized] Incremental Delaunay triangulation algorithm.

Add points one by one, maintaining the Delaunay triangulation. To add a new point \( p \):

1. Find the current triangle \( ABC \) containing \( p \).
2. Join \( p \) to \( A, B, C \).
3. Then flip illegal edges until there are none left.

Issues and details:

1. what if \( p \) is outside the current convex hull?
2. how to limit testing for illegal edges
3. how to find the triangle containing \( p \)
[Randomized] Incremental Delaunay triangulation algorithm.

Issues and details.

1. what if p is outside the current convex hull?

Start by adding a very large outer $\triangle$ s.t. any triangle thru 3 input points has $q_1$, $q_2$, $q_3$ outside.
[Randomized] Incremental Delaunay triangulation algorithm.

Issues and details.

2. how to limit testing for illegal edges after adding point p

Call Test(A,B), Test(B,C), Test(C,A)

where Test(U,V) is a recursive routine to fix edge UV in triangle UVp

Test(U,V) if UV is illegal, flip to pX and call Test(UX), Test(VX).
Changes produced by this Test update:

Some region is retriangulated via a star at p.
All the new edges are incident to p.

Correctness. Why is this limited Test and retriangulation sufficient?

- all the tests and flips we do are correct
- the only issue is that we do not test all the edges to check if they are illegal
Correctness. Why is this limited Test and retriangulation sufficient?

**Claim.** Edges not incident to \( p \) are legal.

**Lemma.** Any edge we add (incident to \( p \)) is legal. In fact, Delaunay.

We flipped to \( px \) because \( p \) inside \( \text{Circle}(uxv) \) and \( p \) is only point inside because we had Delaunay before adding \( p \). Shrink to empty circle through \( x \) and \( p \).
[Randomized] Incremental Delaunay triangulation algorithm.

Analysis of expected run time when points are inserted in random order. Note: we are still ignoring how to find which triangle contains \( p \) (and its runtime).

**Lemma.** The expected time to insert one point is \( O(1) \).

**Proof.**

- Time spent on Test \((u,v)\) = \( O(\# \text{ edges incident to } p) \) at the end.

So we want expected degree of \( p \) in Del. triangulation of \( P_1, P_2, \ldots, P_i = p \), avg. degree in Del. triang. (planar graph) is \( O(1) \).

So backwards analysis gives expected time to insert \( p \) is \( O(1) \).

Total: \( O(n) \) expected # triangles over course of alg. is \( O(n) \).
[Randomized] Incremental Delaunay triangulation algorithm.

**Final issue:** How to find the triangle containing $p$.
The method is easy, the analysis is not.
Note: it is this part of the algorithm that causes the $O(n \log n)$ expected behaviour.

The idea is like Kirkpatrick’s Point Location.
Maintain the history of triangles and changes to them. Then “trace” point $p_i$ through the changes.

- **Two possible triangle updates**
  - Keep blue arrows.
  - Expected space is $O(n)$ because $O(n)$ expected triangles.
  - Each update to locate $p$ costs $O(1)$.
[Randomized] Incremental Delaunay triangulation algorithm.

**Final issue:** How to find the triangle containing p.

How to “trace” p:

- initially (with one big triangle) p is in the big triangle
- at each update, the triangle containing p points to 2 or 3 new ones — check which one contains p

This completes the description of the algorithm.
**Analysis** of expected work to trace $p_i$

Can prove it is $O(\log i)$. Then total expected work to add all points is

$$O(\sum_i \log i) = O(n \log n)$$

First idea: charge work of tracing $p_i$ to each triangle $T$ in the sequence that contains $p_i$

Better idea: charge work to Delaunay triangles that appear in the sequence.

Can show that the expected work for triangles of $D(\{p_1, \ldots, p_j\})$ is $O(3/j)$

$$O\left(\sum_{j=1}^{i-1} \frac{3}{j}\right) = O(\log i) \quad \text{Harmonic series}$$

There is a lovely backwards analysis involved. For details, see [CGAA].
What primitive operations are needed for this algorithm?

Given 4 points, A, B, C, D, is D inside Circle(A,B,C)?

Use the mapping from last day

$$(x, y) \rightarrow (x, y, z = x^2 + y^2)$$

Then the test becomes: is D below the plane through A, B, C?

This is a Sidedness test in 3D, and can be decided with a few multiplications, additions, subtractions.
Summary

- a randomized incremental algorithm for the Delaunay triangulation
- the idea of flipping illegal edges to get to the Delaunay triangulation
- the Delaunay triangulation maximizes the angle vector

References

- [CGAA] Chapter 9.
- [Zurich notes] Chapter 5.