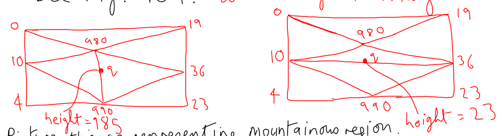
Exam - Each question - 1 algorithm, multiple parts

- Ask about:
  math. concepts arising:
  eg. convex / monotone polygon, Euler Formula
- Describe parts of algorithms (possibly pseudocode)
- Data structures involved
- Running time

pro 13-16:00

Delauney Triangulation

- Consider $f: \mathbb{R}^2 \longrightarrow \mathbb{R}$, whose values we know for finitely many points $P \subseteq \mathbb{R}^2$.
- How to approximate f?
- One way - triangulate the convex hull of P, & define f linearly on each triangle.
  - See Fig. 10.1. What is a good triangulation?



- Picture this as representing mountainous region.
- First case - mountain ridge. Second case - valley.
- Case 1: better approximation.
- Intuitively seems better as no long thin triangles. Triangles have larger angles! We will construct triangulations with large angles.

pro 13-16:08

Prop^n Let P be n points in plane & suppose the convex hull of P has k edges. Then any triangulation of the convex hull has $2n-2-k$ triangles & $3n-3-k$ edges.

Proof m = no. of triangles.
E = no. of edges = no. of edges appearing on 1 triangle (k) + no. appearing on two triangles (ℓ)

Then $3m - \ell = E = k + \ell \Rightarrow 3m = k + 2\ell$.

Euler $m - E + n = 2$

Substituting above equation into Euler gives

$m = 2n - 2 - k$ & similarly

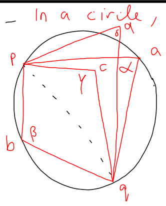
$E = 3n - 3 - k$.

pro 13-16:21

- So any triangulation T of P has m triangles, & so $3m$ angles - we order these angles in a sequence $\alpha(T) = (\alpha_1 \leq \alpha_2 \leq \dots \leq \alpha_{3m})$
- Define $\alpha(T) < \alpha(T')$ lexicographically
  ~ that is, if $\exists j$ s.t. $\alpha_j = \alpha_j'$ for $j < i$ & $\alpha_i = \alpha_i'$.
- Triangulation is angle-optimal if it is maximal in this ordering.
- We will not quite find angle-optimal triangulation, but weaker "legal / DeLauney" triangulation.

pro 13-16:33

- In a circle, $\alpha + \beta = 180°$.

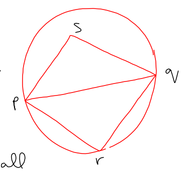


- If c lies inside circle, $\gamma > \alpha$.
- For d outside, $\delta < \alpha$.

See Fig 10.2.

pro 13-16:41

- Consider edge pq in triangulation.
- If pq not on boundary, then we have 2 triangles pqr & pqs.
- Say pq is illegal if s lies strictly inside the circle circumscribing pqr (ie. $\angle psq > 180 - \angle prq$).
- (Equivalently, if r lies inside circle containing pq & s.)
- Otherwise pq is legal.
- A legal triangulation is one in which all edges are legal.



pro 13-16:46

- Given an edge $\vec{pq}$ as above we can "Flip it" to an edge $\vec{rs}$ giving a new triangulation.

Lemma) Let T have illegal edge $\vec{pq}$. Then the flipped edge $\vec{rs}$ is legal in the new triangulation & $\alpha(T) < \alpha(T')$.

See L10.3 & proof in E-Learning - & Fig 10.4.

pro 13-16:55

DeLauney Triangulation

- Another approach.
- Can form Voronoi diagram $V(P)$ from last time:

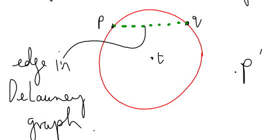
- Delauney Graph is dual graph - same vertices as P (one for each faces of $V(P)$) with an edge from p to $q \iff V(p)$ & $V(q)$ share common edge. See Fig 10.7.

pro 13-17:02

- Can describe DeLauney in elementary terms as follows:
  - From last week, $V(p)$ & $V(q)$ share an edge if $\exists t$ st. $d(t,p) = d(t,q) \leq d(t,p')$ for all other points p' of P.
  - In other words, p & q lie on the boundary of a circle containing no other points of P in its interior.

A DeLauney triangulation is any triangulation of the DeLauney Graph.

10.7 of E-Learning



Theorem) DeLauney Triangulations ≡ legal Triangulations

pro 13-17:11

Algorithm:

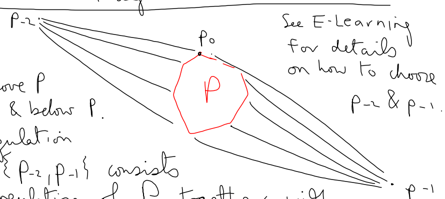
- Could calculate Voronoi diagram of P, calculate its dual & triangulate it.
- We will use legalisation.

Naive version: Find any triangulation of convex hull, go through edges flipping them if illegal.
- Process must stop, since flipping increases position of triangulation in the ordering & only finitely many triangulations.

pro 13-17:19

Randomized incremental algorithm

Step 1)



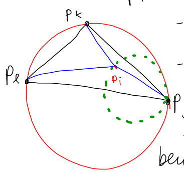
See E-Learning for details on how to choose p_{-2} & p_{-1}.

- p_{-2} is left & above P
- p_{-1} is right & below P.
- legal triangulation of $P \cup \{p_{-2}, p_{-1}\}$ consists of legal triangulation of P together with edges from p_{-2} to each point on left boundary & edges from p_{-1} to each point on right boundary.

pro 13-17:26

Step 2) - Suppose we have legal triangulation T_{i-1} of $P_{i-1} = \{p_{-2}, p_{-1}, p_0, \dots, p_{i-1}\}$ for $i \geq 1$.

- Order of $p_1, \dots, p_n$ is randomised.
- Using search structure D_{i-1} we find a triangle or edge in T_{i-1} where p_i lies.



- Create new triangle as depicted.
- Using that T_{i-1} is Delauney, we see each of p_ip_l, p_ip_k & p_ip_j is legal.
- Some of old edges can become illegal in new triangulation, we must flip these edges, & repeat until process stops.

pro 13-17:32

Step 3) Remove p_{-2}, p_{-1} & all edges connected to them.

Search structure

Oriented graph - leaves are triangle of triangulation.

See E-Learning 10.17, 10.18

- inner nodes are triangles of previous triangulation.

Complexity : expected time $O(n \log n)$.

pro 13-17:42