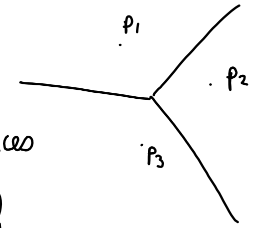
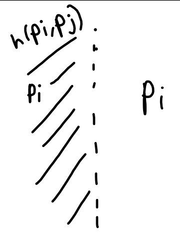
Voronoi Diagrams

Post-office problem:
Consider city with post-offices $P = \{p_1, \ldots, p_n\}$.
Divide city into regions $V(p_i)$ around each post-office such that each point in $V(p_i)$ is closest to p_i.



pro 6-15:40

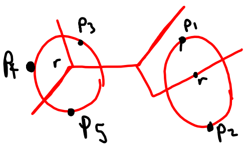
Given $P = \{p_1, \ldots, p_n\}$ in plane,
the Voronoi diagram (V-diagram) is planar subdivision with n faces
$V(p_i) = \{q \in \mathbb{R}^2 : d(q,p_i) \le d(q,p_j) : j \ne i\}$.
Voronoi cell
– Let $h(p_i,p_j) = \{q : d(q,p_i) \le d(q,p_j)\}$
Then $V(p_i) = \bigcap_{j \ne i} h(p_i,p_j)$ & as an intersection of half-planes is a convex polygon.
– By Ch. 5 int. n half-planes takes time $O(n \log n)$. Using this to calc. n voronoi cells $\Rightarrow O(n^2 \log n)$. Today: Faster alg – $O(n \log n)$.



pro 6-16:14

It is not hard to see that:

- $r \in \mathbb{R}^2$ lies on an edge of V-diagram $\Leftrightarrow$ r is equidistant to nearest 2 points of P.
- r is a vertex of V-diagram $\Leftrightarrow$ r is equidistant to nearest 3 points of P.

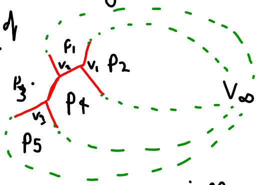


See Fig. 9.3

pro 6-16:23

Theorem
Any V-diagram for set of $n \ge 3$ points (not on a line) has at most $2n-5$ vertices & $3n-6$ edges.

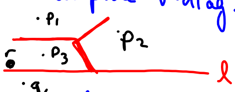
Proof m = no. vertices, h = no. edges of V-diagram.
– Add vertex V_∞ "at infinity" as endpoint for all half-lines.
– Obtain connected planar graph with n faces, m+1 vertices, h edges.
– Satisfies Euler: $(m+1) - h + n = 2$ *
– Degree of vertex ≥ 3, sum of degrees $= 2h$
So $2h \ge 3(m+1)$. Subbing $h = m+n-1$ into * gives $2m + 2n - 2 \ge 3(m+1) \Rightarrow m \le 2n-5$.
Similarly subbing $m = h-n+1$ into * gives $h \le 3n-6$.



pro 6-16:28

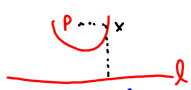
Sweep-line algorithm – Fortune's algorithm (See Youtube)

Would like to use sweep-line approach to compute V-diag. above sweep-line ℓ.
– Problem – new pt. q under sweep line can change the Voronoi-cells above – ie. $r \in V(p_3)$ before ℓ passes q, but $r \in V(q)$ after.
– However, this problem can only occur at a point r for which $d(r,\ell) \le d(r,p)$ for each point p of P above the sweep-line.

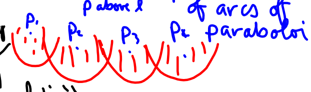


pro 6-16:44

– For p above ℓ, let $\alpha^+(p,\ell) = \{x : d(x,p) \le d(x,\ell)\}$
$\alpha(p,\ell) = \{x : d(x,p) = d(x,\ell)\}$
– By preceding page, we can correctly compute the V-diagram above ℓ in the region $\bigcup_{p \text{ above } \ell} \alpha^+(p,\ell)$
Each $\alpha(p,\ell)$ is a parabola



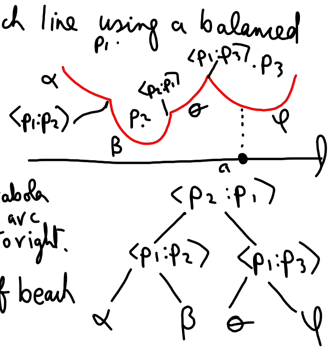
& the union $\bigcup_{p \text{ above } \ell} \alpha^+(p,\ell)$ consists of arcs of parabolae



boundary $\bigcup_p \alpha(p,\ell)$ is "beach line"

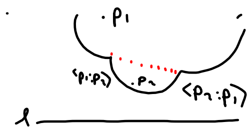
pro 6-16:53

- At ℓ, store beach line using a balanced binary tree:
  - Leaves = arcs of beach line.
  - Internal nodes $\langle p_i : p_j \rangle$ represent "break points" on beach line at which parabola around p_i & p_j meet, with arc of p_i to left & of p_j to right.
  - Given a point a on ℓ, can search tree for arc of beach line above a.



pro 6-17:05

- At breakpoint $r = \langle p_1, p_2 \rangle$ we have $d(r, p_1) = d(r, \ell) = d(r, p_2)$.



- So r is a point on edge of V-diagram.
- If $\langle p_2 : p_1 \rangle$ is on beach line, then it has same distance from p_1 & p_2.
- Therefore the edge from $\langle p_1 : p_2 \rangle$ to $\langle p_2 : p_1 \rangle$ will lie on V-diagram.
- Main technique for constructing edges on V-diagram.

pro 6-17:17

When do new arcs appear or disappear on the beachline?

lemma) A new arc appears on beach line just when sweep-line passes a point of P

- See Fig 9.6 & 9.7

pro 6-17:27

When does an arc disappear?

- See Fig 9.8. Consider 3 consecutive arcs α, β, γ with focuses p_1, p_2, p_3.
- These 3 pts det. a circle with lowest point q & centre s.
- If q lies below ℓ, it is called "circle event" for β.
- Because when ℓ passes q, $d(s,p_1) = d(s,p_2) = d(s,p_3) = d(s,q) = d(s,\ell)$
  so s lies on all 3 arcs.
  In part α & γ meet at s & β disappears.

pro 6-17:33

lemma) An arc of the beach line can disappear only when ℓ passes its circle event.

Algorithm: - Queue Q of events:
  2 kinds - points of P (site event)
  - circle events.
- V-diagram stored in DCEL.
- Binary based search tree T.

- Initially, put all points of P into Q.
  - At first point p, create tree with one leaf & remove p from Q
- When sweep-line crosses site event p_i:
  remove p_i from Q

pro 6-17:41

See E-Learning for full algorithm.

Magic pen not working well today!

pro 6-17:48