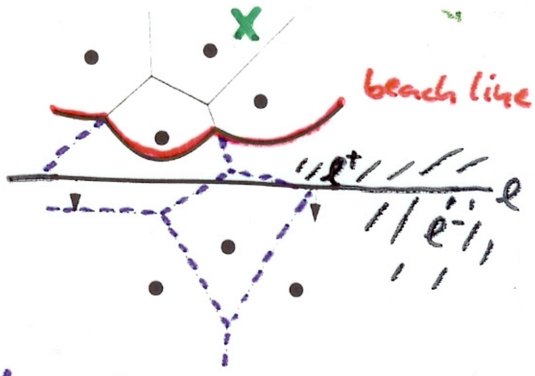
Fortune's Sweep to construct the VD

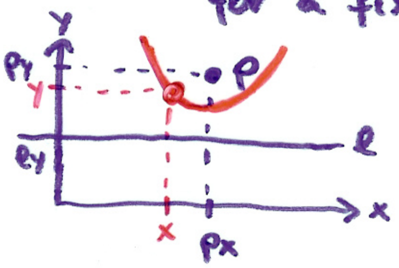
Problem: Cannot maintain intersection of VD with sweepline ℓ since VD above ℓ depends on sites below ℓ

$\Rightarrow$ **Beach Line:** (sweep line status)



- Find points $q \in \ell^+$ for which we know closest site
- If there is a site $p_i \in \ell^+$ s.t. $dist(q, p_i) \leq dist(q, \ell)$
  $\Rightarrow$ closest site of q lies above ℓ
- Beach line := Boundary of set of points $q \in \ell^+$ that are closer to a site above ℓ than to ℓ
  $\rightsquigarrow$ sequence of parabolic ars (why?)
  $\rightsquigarrow$ breakpoints / vertices lie on edges of VD; trace out VD while sweep line moves

Parabola: Set of points (x,y) s.t. $dist((x,y), p) = dist((x,y), \ell)$ for a fixed site $p = (p_x, p_y)$

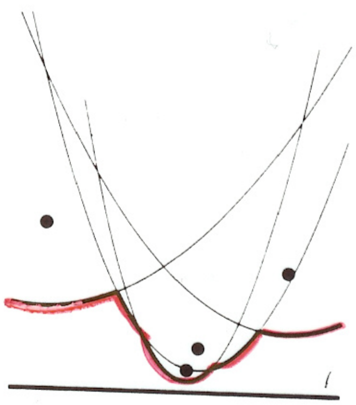


$$(p_x - x)^2 + (p_y - y)^2 = (y - \ell_y)^2$$

$$p_x^2 - 2p_x x + x^2 + p_y^2 - 2p_y y + y^2 = y^2 - 2y\ell_y + \ell_y^2$$

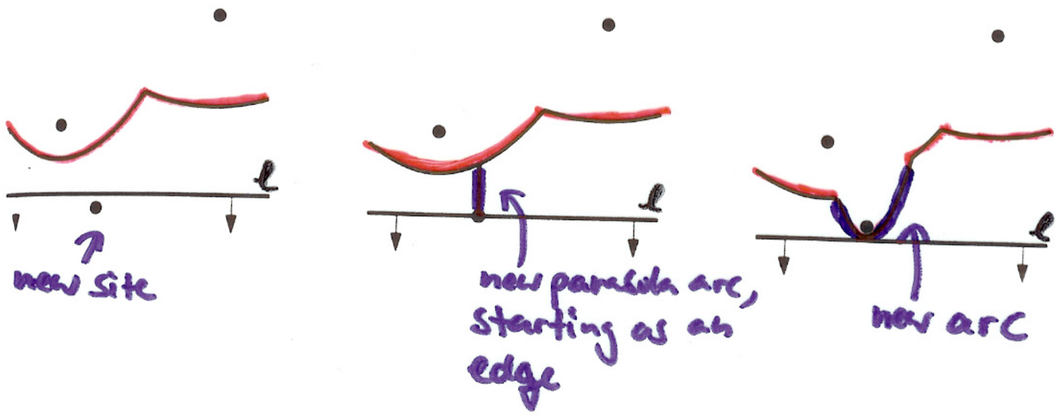
$$y = \frac{p_x^2 - 2p_x x + x^2 + p_y^2 - \ell_y^2}{2(p_y - \ell_y)}$$

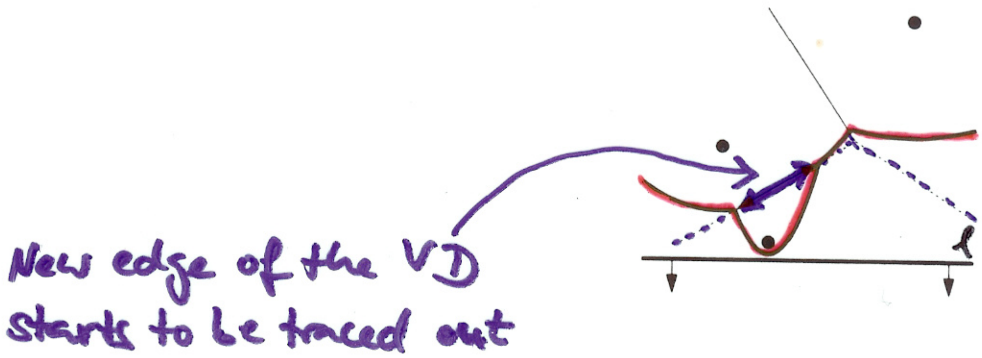
parabola



Site events:

- Sweep line ℓ reaches new site
- (• New parabola in ℓ^+)
- New arc appears on beach line
- New edge of VD starts to be traced out

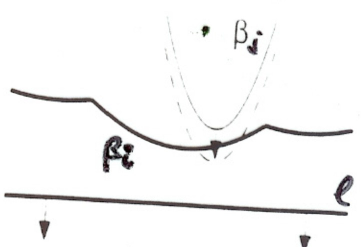




Lemma: The only way in which a new arc can appear on the beach line is through a site event

Proof:

- Assume existing parabola β_j (defined by site p_j) breaks through β_i



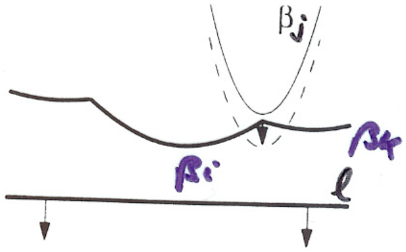
- Formula for parabola β_j:

$$y = \frac{1}{2(p_{jy} - \ell_y)} \cdot \left(p_{jx}^2 - 2\,p_{jx}\,x + x^2 + p_{jy} - \ell_y^2\right)$$

- $p_{jy} > \ell_y$ and $p_{iy} > \ell_y$ $\Rightarrow$ impossible that β_i, β_j have only one intersection point

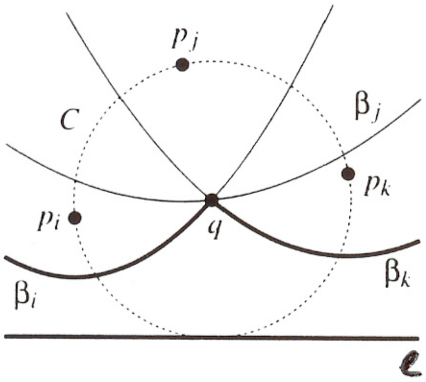
- **2nd case:**

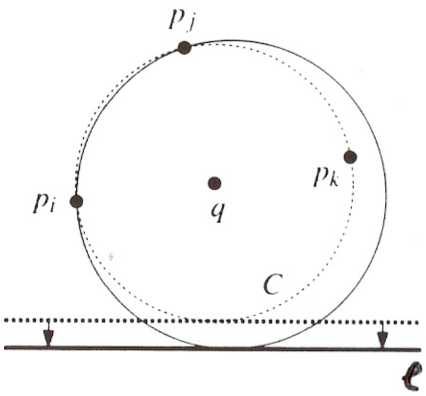
  β_j appears on the breakpoint q between β_i and β_k



- Circle C passes through p_i, p_j, p_k **and** is tangent to ℓ

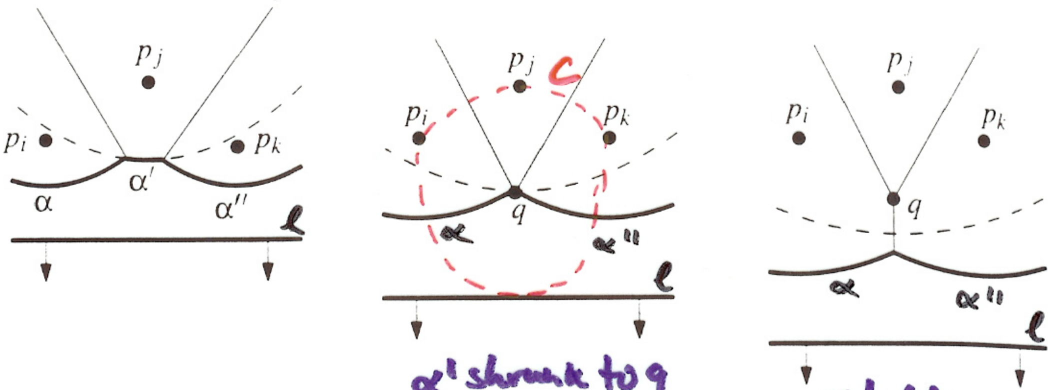
- Infinitesimally small motion of ℓ $\rightarrow$ either p_i or p_k penetrates interior of C
- $\rightarrow$ β_j cannot appear on ℓ





Circle events:

- arc α' shrinks to point q
- arc α' disappears



α' shrunk to q — α' disappeared

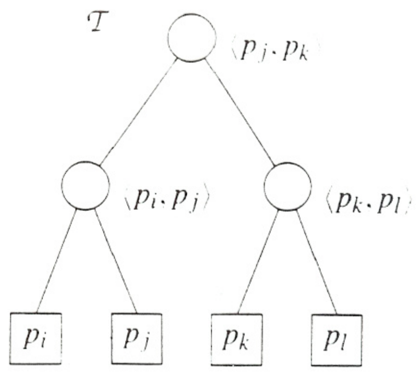
- Circle C passing through p_i, p_j, p_k and touching ℓ (from above)
- No site in interior of C
  (otherwise this site would be closer to q than q is to ℓ, and q would not be on beach line)
- q is VD vertex (two edges of VD meet in q)

Thus: The only way in which an arc can disappear from the beach line is through a circle event

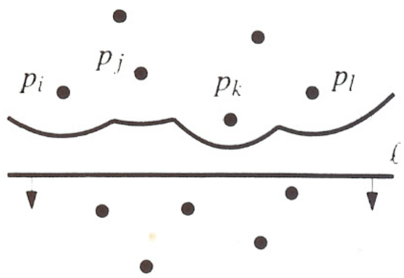
Data Structures

- Store VD under construction in doubly-connected edge list

- Beach line (sweep line status):
  - balanced binary search tree $\mathcal{T}$
  - leaves $\hat{=}$ arcs (in ordered manner)
  - each leaf stores site defining the arc (does NOT store arc; only site)
  - internal node $\hat{=}$ breakpoint on beach line

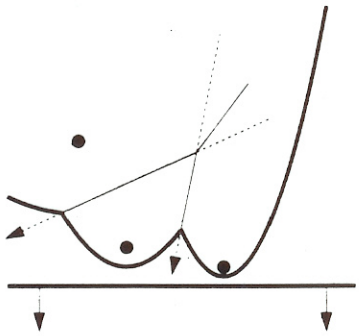


- Event queue Q:
  - priority queue (ordered by y-coord.)
  - site event → store site
  - circle event:
    - store lowest point of circle as event point
    - store pointer to leaf/arc in $\mathcal{T}$ that will disappear



How to detect circle events:

- make sure that for any three consecutive arcs on beach line the potential circle event they define is stored in Q



- Consecutive triples whose breakpoints don't converge → no circle event
- Triple could disappear (e.g., due to appearance of new site) before the event takes place
  → false alarm

Algorithm VORONOIDIAGRAM(P)

Input. A set $P := \{p_1, \dots, p_n\}$ of point sites in the plane.

Output. The Voronoi diagram Vor(P) given inside a bounding box in a doubly-connected edge list $\mathcal{D}$.

1. Initialize the event queue $\mathcal{Q}$ with all site events, initialize an empty status structure $\mathcal{T}$ and an empty doubly-connected edge list $\mathcal{D}$.
2. **while** $\mathcal{Q}$ is not empty
3. **do** Remove the event with largest y-coordinate from $\mathcal{Q}$.
4. **if** the event is a site event, occurring at site p_i
5. **then** HANDLESITEEVENT(p_i)
6. **else** HANDLECIRCLEEVENT(γ), where γ is the leaf of $\mathcal{T}$ representing the arc that will disappear
7. The internal nodes still present in $\mathcal{T}$ correspond to the half-infinite edges of the Voronoi diagram. Compute a bounding box that contains all vertices of the Voronoi diagram in its interior, and attach the half-infinite edges to the bounding box by updating the doubly-connected edge list appropriately.
8. Traverse the half-edges of the doubly-connected edge list to add the cell records and the pointers to and from them.

Degeneracies:

- Two sites same y-coord.
  - → handle in any order
- More than 3 sites on circle
  - Several coincident circle events
  - arbitrary order
  - algorithm produces several degree-3 vertices at same location

Theorem: Fortune's sweep runs in $O(n \log n)$ time and $O(n)$ space.

HANDLESITEEVENT(p_i)

1. If $\mathcal{T}$ is empty, insert p_i into it (so that $\mathcal{T}$ consists of a single leaf storing p_i) and return. Otherwise, continue with steps 2– 5.
2. Search in $\mathcal{T}$ for the arc α vertically above p_i. If the leaf representing α has a pointer to a circle event in $\mathcal{Q}$, then this circle event is a false alarm and it must be deleted from $\mathcal{Q}$.
3. Replace the leaf of $\mathcal{T}$ that represents α with a subtree having three leaves. The middle leaf stores the new site p_i and the other two leaves store the site p_j that was originally stored with α. Store the tuples $\langle p_j, p_i \rangle$ and $\langle p_i, p_j \rangle$ representing the new breakpoints at the two new internal nodes. Perform rebalancing operations on $\mathcal{T}$ if necessary.
4. Create new half-edge records in the Voronoi diagram structure for the edge separating $\mathcal{V}(p_i)$ and $\mathcal{V}(p_j)$, which will be traced out by the two new breakpoints.
5. Check the triple of consecutive arcs where the new arc for p_i is the left arc to see if the breakpoints converge. If so, insert the circle event into $\mathcal{Q}$ and add pointers between the node in $\mathcal{T}$ and the node in $\mathcal{Q}$. Do the same for the triple where the new arc is the right arc.

(· New site could be left, middle, or right of a triple
· middle arc → left & right come from same para → diverge)
O(log n) time per event; n events

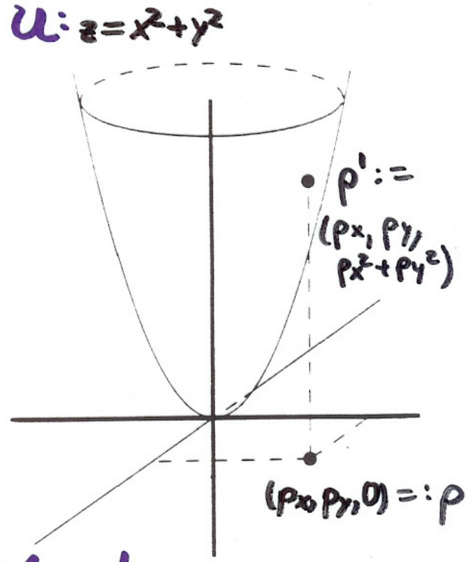
HANDLECIRCLEEVENT(γ)

1. Delete the leaf γ that represents the disappearing arc α from $\mathcal{T}$. Update the tuples representing the breakpoints at the internal nodes. Perform rebalancing operations on $\mathcal{T}$ if necessary. Delete all circle events involving α from $\mathcal{Q}$; these can be found using the pointers from the predecessor and the successor of γ in $\mathcal{T}$. (The circle event where α is the middle arc is currently being handled, and has already been deleted from $\mathcal{Q}$.)
2. Add the center of the circle causing the event as a vertex record to the doubly-connected edge list $\mathcal{D}$ storing the Voronoi diagram under construction. Create two half-edge records corresponding to the new breakpoint of the beach line. Set the pointers between them appropriately. Attach the three new records to the half-edge records that end at the vertex.
3. Check the new triple of consecutive arcs that has the former left neighbor of α as its middle arc to see if the two breakpoints of the triple converge. If so, insert the corresponding circle event into $\mathcal{Q}$, and set pointers between the new circle event in $\mathcal{Q}$ and the corresponding leaf of $\mathcal{T}$. Do the same for the triple where the former right neighbor is the middle arc.

- O(log n) time per event
- Each processed event defines Voronoi vertex (O(n) many)
False alarms are deleted before they are processed

Voronoi diagrams and halfspace intersections

- $\mathcal{U} := (z = x^2 + y^2)$ unit paraboloid
- Consider the VD to be embedded in the plane $z = 0$
- Let $p := (p_x, p_y, 0)$
  and $p' := (p_x, p_y, p_x^2 + p_y^2)$ vertically above p on $\mathcal{U}$

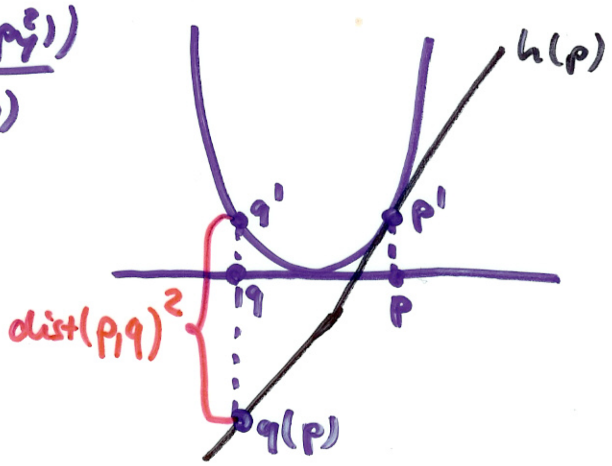


- Let $h(p): z = 2p_x x + 2p_y y - (p_x^2 + p_y^2)$ plane
  - $h(p)$ contains p'
  - $h(p)$ is tangent to $\mathcal{U}$ at p'
- Let $H(P) := \{ h(p) \mid \text{site } p \in P \}$
- Consider convex polyhedron

$$\mathcal{P} := \bigcap_{h \in H(P)} h^+ \quad ; \quad h^+ := \text{halfspace above } h$$

<u>Claim:</u> The projection of the edges and vertices of $\mathcal{P}$, vertically downwards to $z = 0$, is the $VD(P)$

- $q(p) := (q_x, q_y, \underbrace{2p_x q_x + 2p_y q_y - (p_x^2 + p_y^2)}_{= (q_x^2) + (q_y^2) - \mathrm{dist}^2(q, p)})$



- Let $q \in$ Voronoi cell of p
  $\leadsto \mathrm{dist}(q, p) < \mathrm{dist}(q, r)$
  for all $r \in P$, $r \neq p$
- Of all points in P,
  p has smallest distance to q
  $\leadsto q(p)$ is highest intersection point
  $\leadsto$ vertical line through q intersects highest possible plane $h(p)$