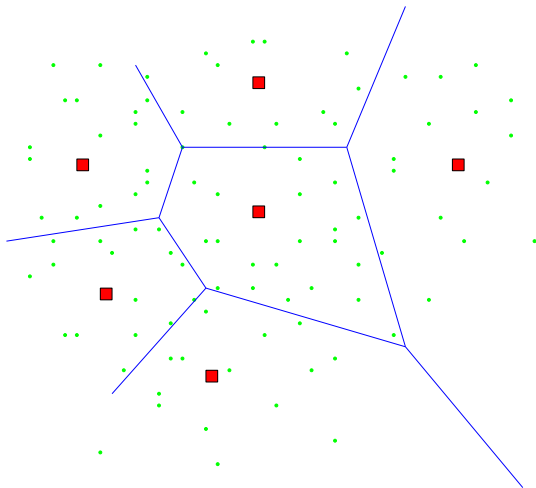


Voronoi Diagrams

Voronoi Diagrams – Motivational Example

The Post Office Problem

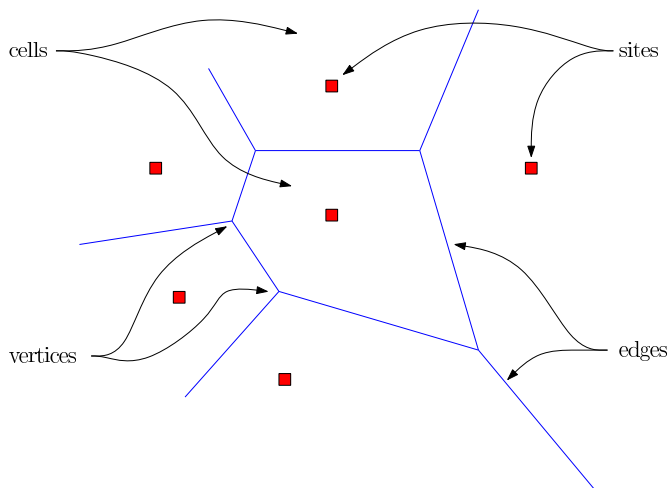
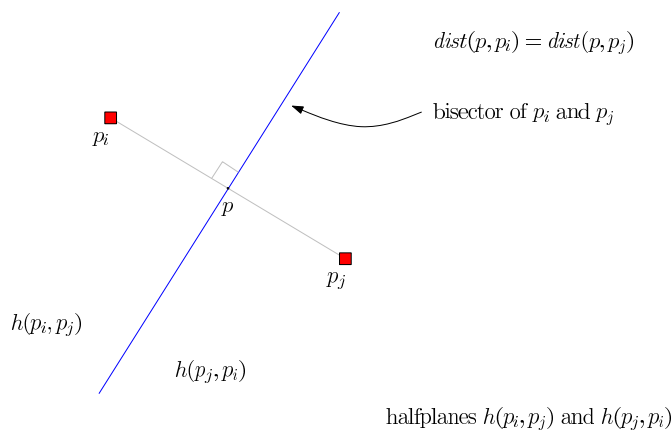


Voronoi Diagrams – Definition and Terminology

Definition and Terminology

Definition:

The Voronoi Diagram $Vor(P)$ of a set $P = \{p_1, \dots, p_n\}$ of n sites is the subdivision of the plane into n cells, one for each site in P , such that a point q lies in the cell for p_i iff $dist(q, p_i) < dist(q, p_j)$ for all $p_j \in P$ with $i \neq j$.



Basic Properties

- cell of $p_i = \bigcap_{1 \leq j \leq n, j \neq i} h(p_i, p_j)$
- cell of p_i is open convex region

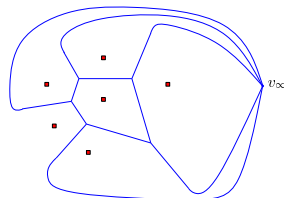
- **Theorem:**
Let P be a set of n sites in the plane. If all sites are collinear then $Vor(P)$ consists of $n - 1$ parallel lines. Otherwise, $Vor(P)$ is connected and its edges are segments or half lines.

- the size of $Vor(P)$ is at most quadratic, because:
we have at most n cells and each cell has $\leq n - 1$ edges, but:

- **Theorem:**
For $n \geq 3$, the number of vertices n_v in $Vor(P)$ of a set P of n sites is $\leq 2 \cdot n - 5$ and the number of edges n_e is $\leq 3 \cdot n - 6$.

Proof:

We use Euler's formula for connected plane graphs:
 $v - e + f = 2$.

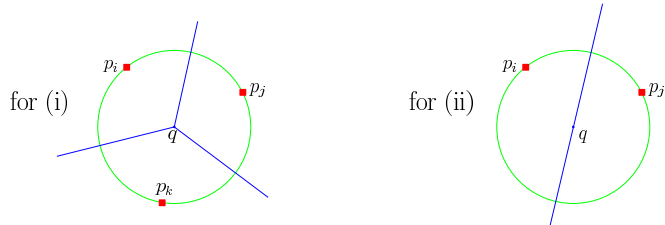


Let q be a point. The largest empty (with respect to P) circle $C_P(q)$ is the largest circle with q as centre that does not contain any site of P in its interior.

Theorem:

Let be given a set P of sites and $Vor(P)$.

- (i) Point q is a vertex of $Vor(P) \iff C_P(q)$ contains ≥ 3 sites on its boundary.
- (ii) The bisector between p_i and p_j defines an edge of $Vor(P) \iff$ there is a point q on the bisector, such that $C_P(q)$ contains both p_i and p_j on its boundary, but no other site.



Computing $Vor(P)$

There is an algorithm to compute the intersection of n half planes in $O(n \log n)$ time.

$\Rightarrow Vor(P)$ can be computed in $O(n^2 \log n)$ time

However, with a plane sweep algorithm, called Fortune's algorithm, we can compute $Vor(P)$ in $O(n \log n)$ time.

Faster is not possible, because sorting can be reduced to computing $Vor(P)$.

\Rightarrow Fortune's algorithm is optimal

Problem (when sweeping from top to bottom):

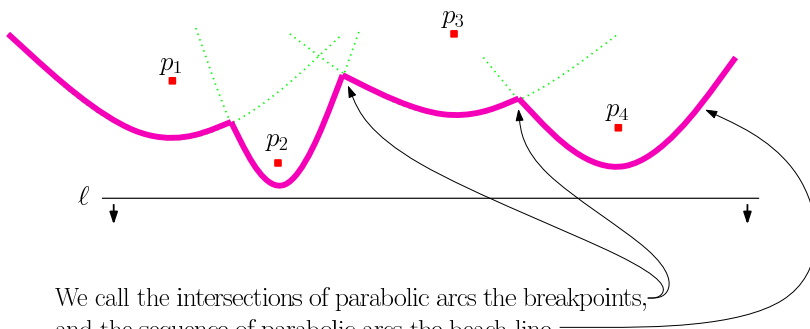
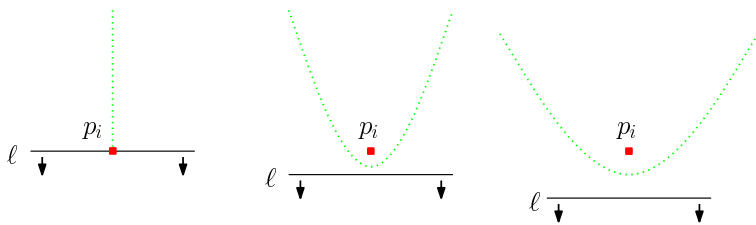
The part of $Vor(P)$ above the sweep line ℓ depends partially also on sites below ℓ .

Solution:

We do not maintain the intersection of ℓ and $Vor(P)$, but we do maintain information about the part of $Vor(P)$ of sites above ℓ that cannot be changed anymore by sites below ℓ .

Observations:

- If $dist(q, p_i) < dist(q, \ell)$ with p_i above ℓ , then q belongs to some cell of a site above ℓ .
- The locus of points that are closer to some site p_i above ℓ than to ℓ is bounded by a parabola.



We call the intersections of parabolic arcs the breakpoints, and the sequence of parabolic arcs the beach line.

Observations:

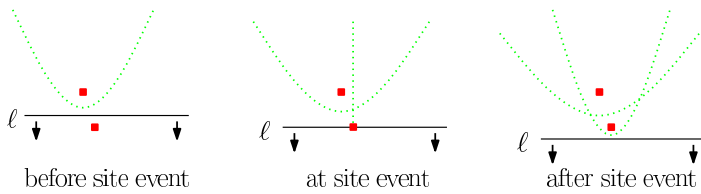
- The beach line is x -monotone.
- The break points lie on edges of $Vor(P)$.

Idea:

Maintain the beach line during the sweep (but not explicitly).

The sweep is from top to bottom, stopping at:

- site events
 - ℓ hits a site
 - precomputed according to y -coordinates of sites
- circle events
 - an existing arc of the beach line shrinks to a point and disappears
 - computed during the sweep



Recall: Breakpoints trace out the edges of $Vor(P)$.

A new parabolic arc appears on the beach line.

Two new breakpoints appear (they coincide at the beginning).

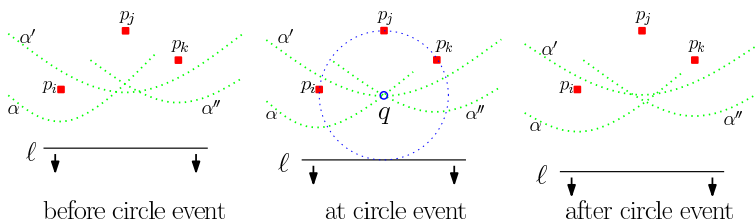
They trace out a new edge (initially not connected to $Vor(P)$).

Lemma:

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

Consequence:

The beach line consists of $\leq 2 \cdot n - 1$ parabolic arcs, because:
 each site creates ≤ 1 new arc
 and ≤ 1 splitting of another arc into 2.



Observations:

- α and α'' cannot be part of the same parabola
 - the arcs are defined by 3 distinct sites: p_i, p_j and p_k
- Let q be the point the 3 arcs pass through when α' disappears.
- the circle passing through p_i, p_j and p_k , centred at q touches ℓ
 - no other site can be in that circle
 - q is a vertex of $Vor(P)$

Lemma:

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

But how to detect circle events?

At every event, the algorithm checks all the new triples of consecutive arcs.

When converging break points are discovered, insert corresponding circle event into event queue.

For all disappearing triples, check if they have an event in the queue, if so delete it, because it is a false alarm.

Lemma:

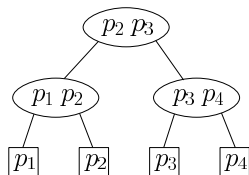
Every Voronoi vertex is detected by a circle event.

Datastructures:

- balanced binary search tree T
 - event queue Q
 - a data structure to store $Vor(P)$
-

Balanced Binary Search Tree T :

- represents the beach line (it is the status structure)
- leaves store sites (not the arcs) in an ordered way
- internal nodes represent break points on the beach line
- allows operations in $O(\log n)$ time
- contains pointers to the other data structures



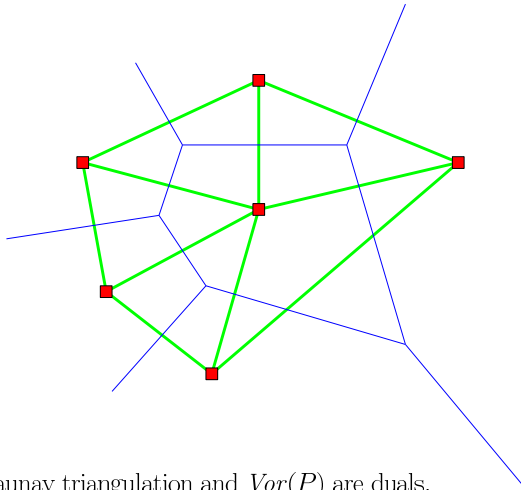
Event Queue Q :

- priority queue according to y -coordinates of events
- site events are a priori known and stored in Q
- circle events: we store the lowest point of the circle and a pointer to the corresponding leaf in T

After the algorithm Q is empty, but T perhaps not.
The still existing break points correspond to the half infinite edges of $Vor(P)$.

Main Theorem:

The Voronoi Diagram of a set of n sites in the plane can be computed with a sweep line algorithm in $O(n \log n)$ time using $O(n)$ space.



Delaunay triangulation and $Vor(P)$ are duals.
