[Voronoi Diagrams]

Computational Geometry (CSc 635)

Jagdish Bhatta

Central Department of Computer Science & Information Technology

Tribhuvan University

Voronoi Diagrams

(1850 by Dirichlet & by Voronoi in 1908)

Voronoi Polygons:

Let P = {p_1, p_2, p_3, ------, p_n} be a set of points in the plane, where each p_i is considered as point site. The Voronoi polygon for each point site pi denoted by V (p_i) is a convex polygon that encloses all the points at least closer to p_i, then any other point sites.

i.e. $V(p_i) = \{q / |p_i - q| \leq |p_j - q| \ \forall j \neq i\}$

i.e. $V(p_i) = \{q / \text{distance}(q, p_i) \leq \text{distance}(q, q_j), \forall j \neq i \in P\}$

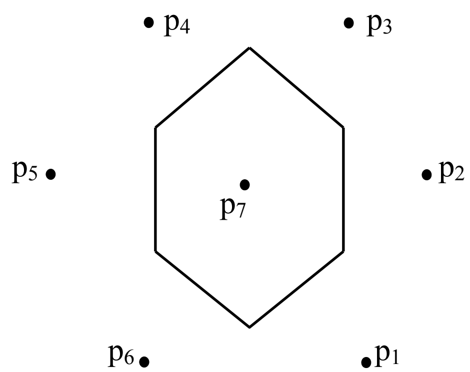


Fig:- Voronoi polygon of $p_7 \approx V(p)$

Thus we can define a Voronoi diagram induced by n-point sites as a collection of n-Voronoi polygons corresponding to each point sites, contributing to one Voronoi polygon.

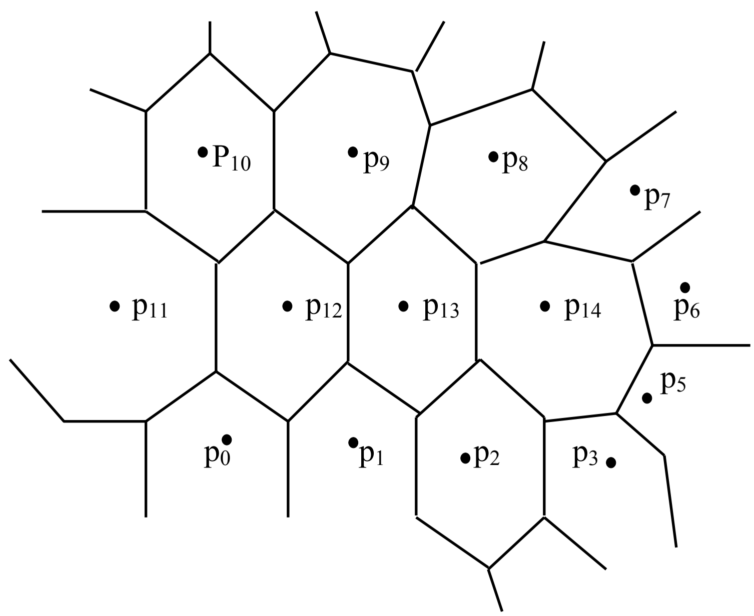
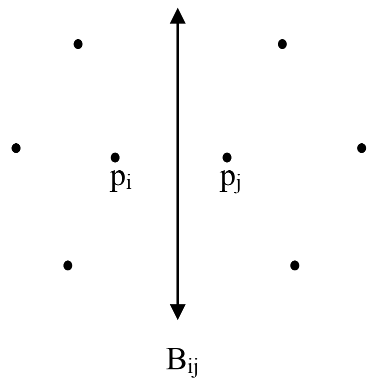


Fig:- Voronoi Diagram of 15 point sites

Interpreting in terms of half planes:

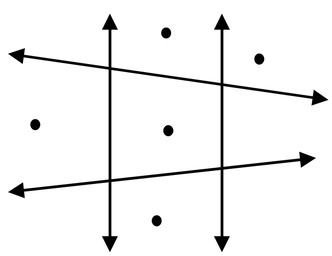
Given n point sites p_1, p_2, p_3, -------, p_n. Let H (p_i, p_j) be a closed half plane with the bisector B_{ij} as its boundary and containing p_i.

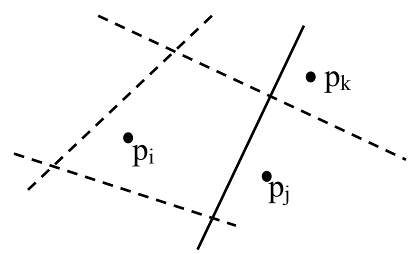


Then, H (p_i, p_j) can be viewed as all the points that are closer to p_i than they are to p_j we know V (p_i) is set of all points closer to p_i, than to any other site i.e. points closer to p_i than to p_1 and points closer to p_i than to p_2 and points closer to p_i than to p_3 and so on.

Thus, the Voronoi polygon can be written as intersection of all half planes & written as;

$$V(p_i) = \bigcap_{i \neq j} H(p_i, p_j)$$





The Voronoi region of any point set could be bounded or unbounded.

Voronoi Edge: - The edges of Voronoi diagram are the Voronoi edges.

Voronoi Vertex: - All the vertices of Voronoi diagram are called the Voronoi vertices.

Thus, the Voronoi diagram of n-point sites partitions the plane into n-convex region such that the region corresponding to p_i consists of all points closer to p_i, then any other point sites p_j. If the Voronoi regions are bounded, they are convex polygons.

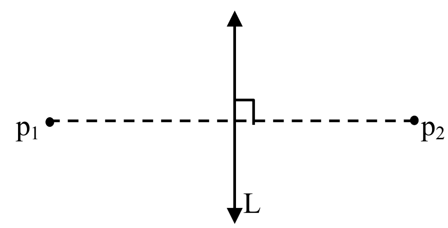
Voronoi Diagram for Simple Cases: -

- *Voronoi diagram with single point site:*

  • p_1

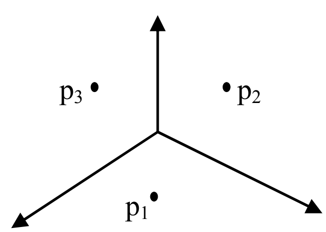
  Here, the Voronoi diagram consists of a single region containing all the points in plane.

- *Voronoi diagram with two point sites:*



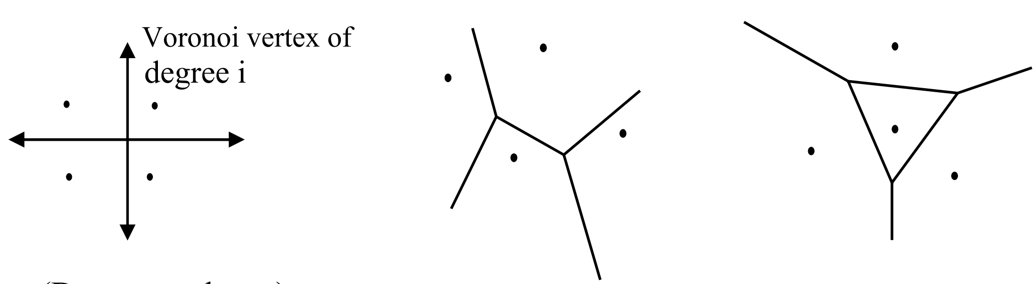
  The Voronoi diagram for a set of points sites $P = \{p_1, p_2\}$ consists of two half planes corresponding to each point site pi. The half planes are divided by a line segment L, which is a perpendicular bisector of the segment p_1p_2.

- *Voronoi diagram with three point sites:*



  The Voronoi diagram of a set of point sites, $P = \{p_1, p_2, p_3\}$ consists of three Voronoi regions, where all regions are unbounded.

- *Voronoi Diagram with four point sites:*



(Degenerated case)

(i) (ii) (iii)

The Voronoi diagram for a set of point sites P = {p_1, p_2, p_3, p_4} consists of four Voronoi regions, all of which may be unbounded or one of them may be bounded & rest three unbounded.

The figure (i) above is degenerated case where all points are concyclic. In this case, there will be a single Voronoi vertex with degree 4. But in general, each Voronoi vertices are of degree 3.

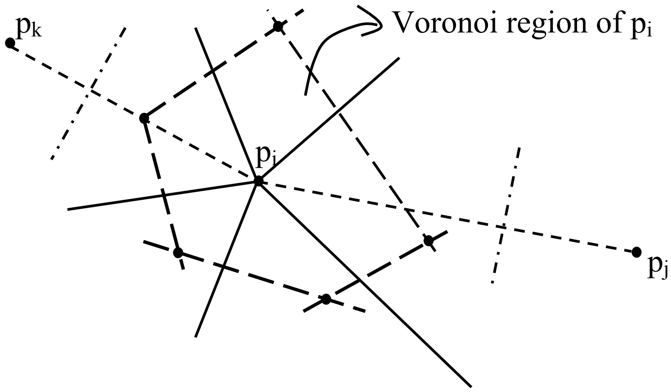
Application of Voronoi Diagram:

- *Fire observation towers:* Consider a forest with a number of observation towers. Each ranger is responsible for extinguishing any five closer his tower than to any other tower. The set of all trees for which a particular ranger is responsible constitutes the Voronoi polygon associated with his tower.
- *Nearest neighbor queries:* Given a set of points P and a query point q, determine a point p that is closest to q. for this we can compute the Voronoi diagram & then locating cell of the diagram that contains q, we can produce the result.
- *Facility location:* If we want to open a new video store, then it will be better to be placed as far as possible from any existing video store, for best selling. It is equivalent to locating the new store at the center of largest empty circle, whose interior contains no other stores.
- Path Planning,
- Crystallography,
- Computational Morphology & so on.

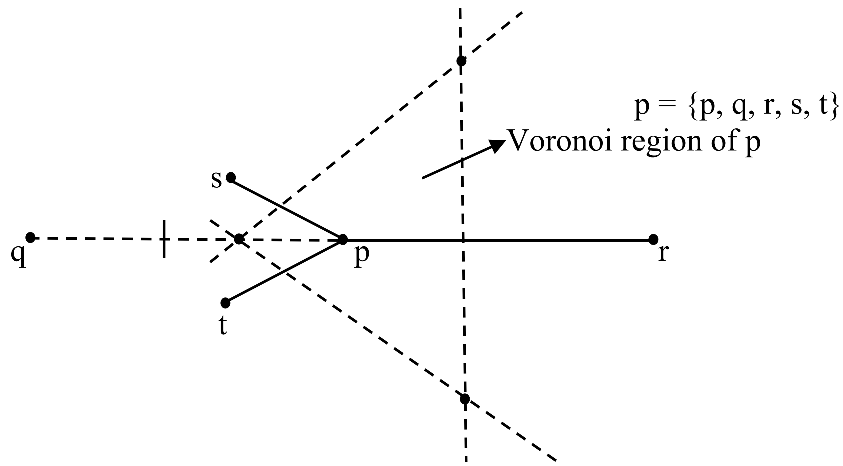
Construction of Voronoi Diagram:

- A simple approach for constructing a Voronoi region for a given point site p_i in the set P of point sites, is to take all the perpendicular bisectors of the segments connecting p_i to remaining members of P. These bisectors will delimit the half planes containing point site. The intersection of all half planes containing p_i is the Voronoi region of p_i.

- At first, we can start with the segments connecting p_i to all remaining members of P. then gradually extend the lines outwards along the perpendicular bisector of these segments until they intersect.



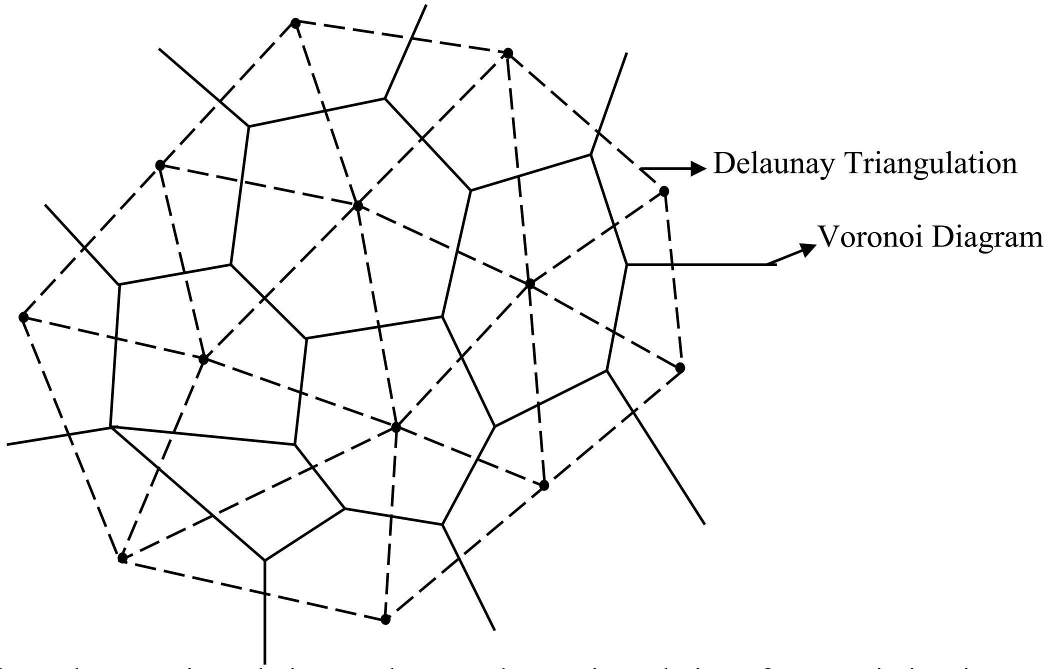
The point's p_k & p_j do not contribute for Voronoi region of p_i.



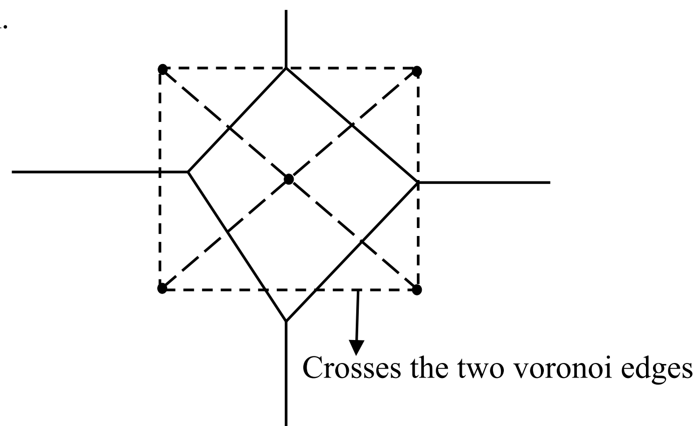
From this figure, we can conclude that not only farthest, even though a point site is nearer to another point site, it may not contribute to the Voronoi region for next point site. And also a point being farthest enough from a point site may contribute to its region.

Delaunay Triangulation (Dual of Voronoi Diagram):

The dual of Voronoi diagram/Delaunay triangulation is a graph G having its nodes corresponding to each Voronoi cell (polygon) equivalently every point site, and an edge (arc) between two nodes if the corresponding Voronoi polygons share an edge.



- The Delaunay triangulation produces a planar triangulation of Voronoi pint sites.
- It may be case that the edge of Delaunay triangulation may cross two edges of Voronoi diagram.



Properties of Delaunay Triangulation:

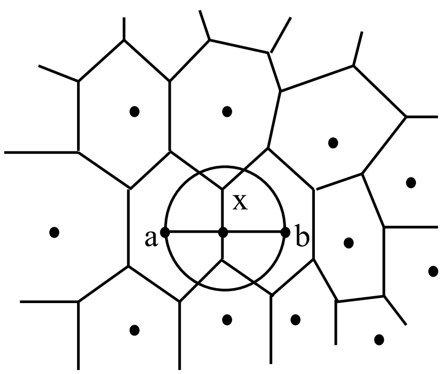
1) Delaunay triangulation is the straight-line dual of Voronoi diagram.
2) Every face of Delaunay triangulation is a triangle, if no four points are co circular.
3) Each face of Delaunay triangulation corresponds to a vertex of Voronoi diagram.
4) Each edge of Delaunay triangulation corresponds to an edge of Voronoi diagram.
5) Each node of Delaunay triangulation corresponds to a region of Voronoi diagram.
6) The boundary of Delaunay triangulation is convex hull of the sites in the Voronoi diagram.
7) Interior of each (triangle) face of Delaunay triangulation contains no sites.

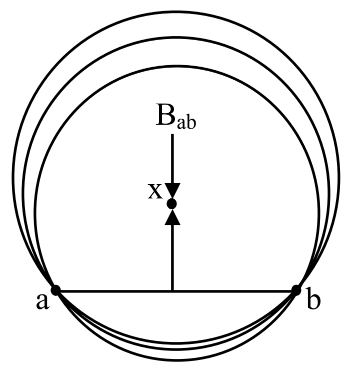
Properties of Voronoi Diagram:

1) Each Voronoi region V (p_i) is convex.
2) V (p_i) is unbounded if and only if p_i is on the convex hull of the point set.
3) If v is a Voronoi vertex at the function of V (p_1), V (p_2) & V (p_3), then v is the center of the circle C (v) determined by p_1, p_2 & p_3. (It holds true for Voronoi vertex of any degree)
4) The circle C (v) is circum- circle of the Delaunay triangle corresponding go v.
5) The interior of C (v) contains no other point sites. (empty circle)
6) If p_j is nearest neighbor of p_i, then (p_i, p_j) is an edge of Delaunay triangulation.

Theorem: Let P is a set of point sites with a, b ∈ P then ab ∈ D (P) if and only if there is an empty circle through a & b.

Proof:





(If part)

If ab ∈ D (P), then V (a), Voronoi region of a, & V (b), Voronoi region of a, share an edge e ∈ V (P).

Let C (x) is a circle with center x on the interior of e with ax = bx and x ∈ V (a) and x ∈ V (b) only. Clearly, this circle is empty of other sites. If not, then there is a site C or in the circle then, x ∈ V (C) as well. But we know x is only in V (a) & V (b) so site C can't be on or in C (x). Hence, C (x) is empty.

(Only if Part)

Let C (x) is an empty circle passing through the point sites a & b, having x as its center. Clearly, x is equivalent from a & b so, x ∈ V (a) and x ∈ V (b) as long as no other point interferes with nearest neighborliness. But the circle C (x) is empty, & it passes though only two point sites a & b, so no other points interferes.

Hence, $x \in V(a) \cap V(b)$

Because no points are on the boundary of C (x), other than a & b (by hypothesis), we can maintain emptiness by moving x along B_{ab}, the bisector of ab, keeping the circle through a & b. therefore, x is on positive length Voronoi edge shared between V (a) & V (b).

$\therefore$ ab $\in$ D (P)

Computing the Voronoi diagram:

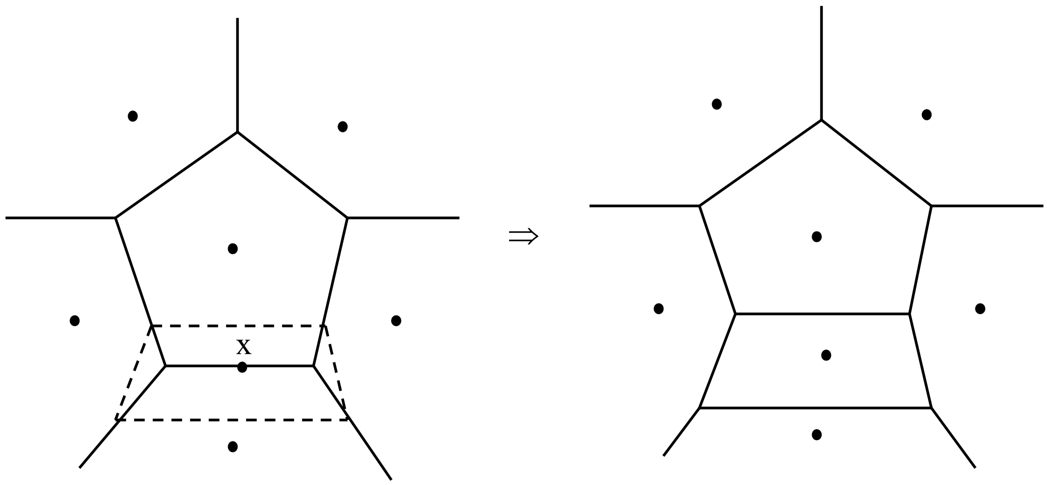
There are a number of algorithms for computing Voronoi diagrams. The naïve approach that takes $O(n^2 \log n)$ time operates by computing $V(p_i)$ by intersecting the n-1 bisector half planes $h(p_i, p_j)$ for $j \neq i$. there are also the incremental algorithms for constructing the Voronoi diagrams & take $O(n^2)$ time.

When computational geometry came along, more complex, but asymptotically superior O(nlogn) algorithms based on divide & conquers, approach was discovered, but it was rather complex & somewhat difficult to understand later on. Fortune's algorithm based on the idea of plane sweep approach was invented by Steven Fortune. It provided a simpler O(nlogn) solution for the problem.

Incremental Approach for Computing Voronoi Diagram:

It constructs the Voronoi diagram by updating the existing Voronoi diagram after inserting a new point, thus, incremental.

It constructs the voronoi diagram by updating the existing voronoi diagram after inserting a new point, thus, incremental.

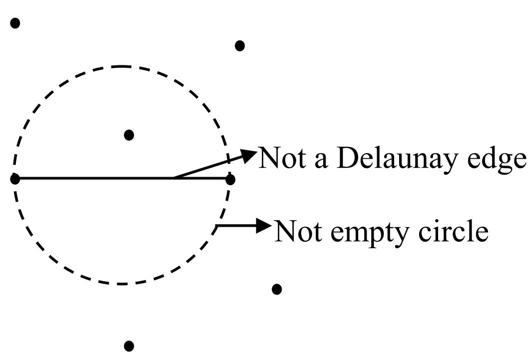


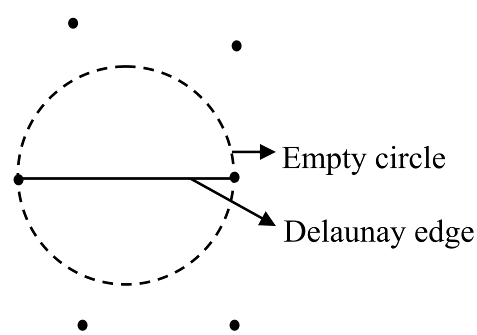
Consider a Voronoi diagram of i point sites (i = 6 in this case) constructed at certain instant as shown above. If we insert $(i+1)^{th}$ point, say x in above figure, then this point may lie within the circles corresponding to many Voronoi vertices. This violates the condition that Voronoi vertex circle must be empty of sites.

Thus, after inserting new point site x, the Voronoi diagram needs to be updated. Here, only those portions of Voronoi diagram are updates which are overlapping with the circles that are containing x within it. Here, each update takes O(n) time & there at most O(n) such updates. Hence, total complexity is $O(n^2)$.

In Circle Test (Direct Characterization of Delaunay Triangulation): -

Given a set S of point sites; two point sites a, b∈S are the end points of Delaunay edge if and only if there a circle through a & b that passes through no other point sites and contain no other point sites within it i.e. there is an empty circle through a, b. this is known as In-Circle test. Thus, determining the Delaunay edge can be done through In-Circle test.
Thus, determining the Delaunay edge can be done through In-Circle test.





Using the In-Circle test Delaunay triangulation of n point sites can be constructed. If there is an empty circle passing through two point sites then an edge for Delaunay triangulation is drawn otherwise there will not be the Delaunay edge between the two point sites.

Thus, proceeding in such a way, Delaunay triangulation can be obtained without obtaining the Voronoi diagram of the point sites.

Largest Empty Circle Problem:

Given a set of point sites S in plane, find the largest empty circle that contains no points. Since the circle having its center outside the hull of point sites can be made arbitrarily large in the region of plane, so a restriction to the center of circle, in its location, will make the problem having wider sense. So, we can phrase the problem as;

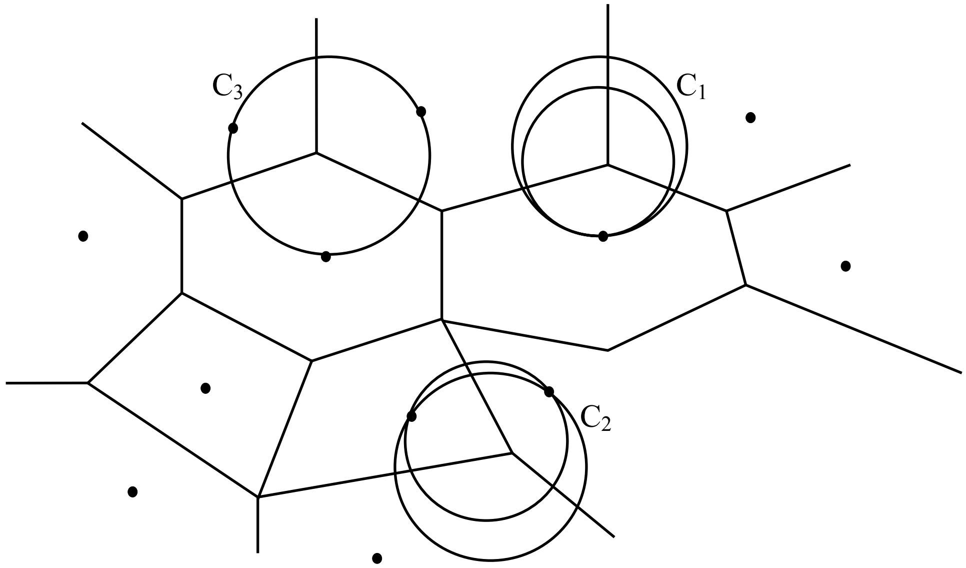
- Find a largest empty circle whose center is in the convex hull of point sites. It is empty in that it contains no sites in its interior & largest in that there is no other such circle with strictly larger radius.

So, interpreting this problem now finding the largest empty circle, we may have two cases:

- The center of circle lies inside the hull
- The center of circle lies on the hull

Case I: Centers inside the hull:

- If the circle includes just one site, then it can't be the largest circle, it can be made further larger by adjusting the circle. As the circle C_1, in figure:



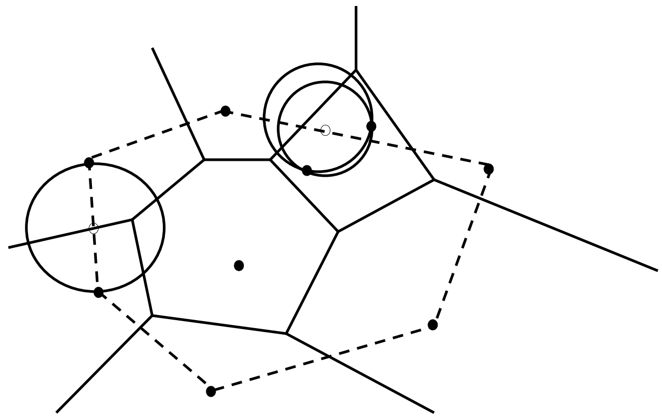
Similarly circle C_2 with two point sites can be made larger by adjusting circle.

- If the circle includes there point sites on its boundary, then it is the largest one as it can't be made larger since the three points span more than a semicircle. So adjusting center of circle will make the circle to inscribe any one point site. Hence, any circle C_3 in the figure above.

Case II: Centers on Hull:

As in previous case, we can argue the maximum radius circle with center on hull.

- If circle includes one point site on its boundary, it can be made larger by adjusting the center along the edge of the hull.
- If circle contains two point sites, then it will be the maximum largest circle, as adjusting center may either inscribe point site within circle or, the center may go outside the hull. So if circle contains any two point sites then it is the largest one.



From the above analysis, we can have following algorithm for computing largest empty circle;

- Find the circles corresponding to each Voronoi edge that lie on interior of convex hull of point sites.
- Find the circles corresponding to the point of intersection of convex hull and Voronoi edges.
- Report the largest circle among all the circles obtained in above steps.

For these steps, compute as follows:

- First compute Voronoi diagram of set of point sites, S.
- Compute convex hull of point sites H = H (S).
- For each Voronoi vertex v do

    if v is inside H then

        compute radius of circle, centered at v and update maximum radius.

    end if

  end for

- For each Voronoi edge e do

  Compute intersection point, p of edge e with hull boundary of H.

  Compute radius of circle centered at p and update maximum radius.

  end for
- Return the maximum radius. So the circle with maximum radius will be the largest empty circle.

Nearest Neighbor Problem:

- Gives a point site p_i, find the nearest neighbor (site) of p_i. The nearest neighbor relation among a set points p can be defined as;

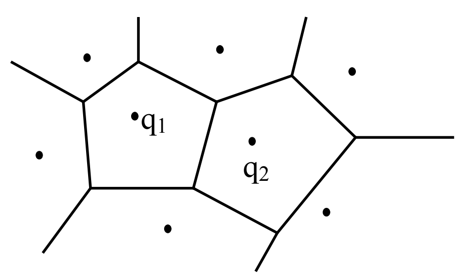
  For point a & b∈P, b is a nearest neighbor of b if and only if $|a–b| \leq \min|a–c|$ for all c∈P & a≠c.
- It can be identified as; a→b: A nearest neighbor of a is b.
- Nearest neighbor is not a symmetric relation i.e. if a→b, it is not necessary that b→a.
- A point can have several equally nearest neighbors.

Computing Nearest Neighbor:

Voronoi diagram can be used to find the nearest neighbor of a point site. If a fixed set of points P is given, constructing Voronoi diagram of P, we can obtain the nearest neighbor relationships.

Thus, for a point site pi, if V (p_i) is the Voronoi polygon corresponding to p_i, then nearest neighbor of p_i is one of the incident sites corresponding to pi, new nearest neighbor of pi one of the incident site corresponding to the Voronoi polygon v (p_i).

Now for a query point q, finding the nearest neighbor of q reduces the problem to finding in which Voronoi region q lies (falls). The site corresponding to this region is nearest to q.



The algorithm can be traced as;

- Complete the Voronoi diagram & store it in DCEL representation.
- For each point site pi do

For each edge of V (p_i) do

Identify the neighbors of p_i inspecting all edges of V (p_i) & corresponding sites. (Neighbors)

end for

Output the identified nearest neighbors.

end for

(Note: Nearest neighbor relationships can be used in pattern recognition, molecular biology etc)

Nearest Neighbor graph (NNG):

It is such a graph whose nodes are associated with each point site, from set of point sites P, and an arc between them if one point is a nearest neighbor of the other. Since, nearest neighbor relation is not symmetric, so NNG is always a directed graph.

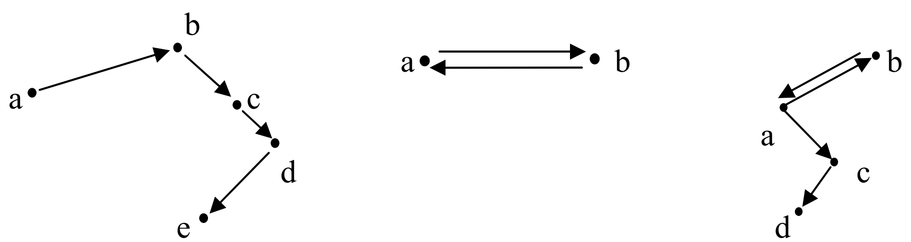


Fig:- Nearest Neighbor Graph

A nearest neighbor graph of point sites is the subset of Delaunay triangulation of the point sites. i.e. NNG $\subseteq$ DT

Euclidean Minimum Spanning Tree (EMST):

A minimum spanning tree (MST) of a set of points S, in the plane is the shortest weight or minimum length tree whose nodes are all points in S. When the length of an edge from MST is measured using the Euclidean length of the segment connecting its end points, then the tree is known as Euclidean Minimum Spanning Tree (EMST). EMST have many applications as in networking areas where local area networks take the form of a tree spanning the host nodes. Thus, MST minimizes the wire lengths which reduces cost & time delays. To compute EMST, we can use Kruskal's or Prim's algorithm.

Kruskal's Algorithm for Computing EMST:

⇒ Given n point sites, compute EMST.

Let T be a tree constructed incrementally during the computation. T+e is a tree after adding edge e into T. Now, Kruskal's Algorithm is as:

- Sort all edges of graph by length in ascending order. Let the sorted list be e_1, e_2, ---- up to O (n^2) edges.
- Initialize T = ϕ
- while T is not spanning i.e. number of edges in T is less than n-1 (T! = n-1)

  if T + e_i is forms cycle.

  Discard e_i

  else

  T = T + e_i

  i = i + 1

  end while

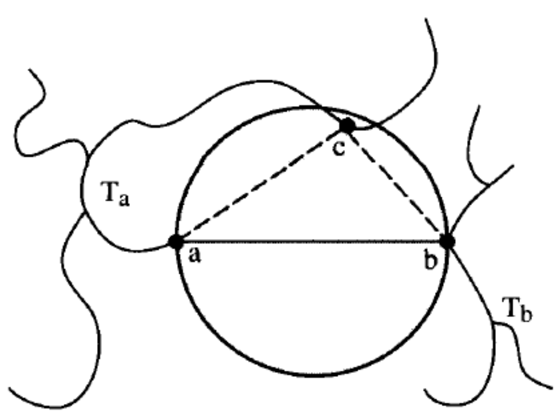
Clearly, there are at most $\binom{n}{2}$ edges i.e. O(n^2) edges. So sorting those it requires O(n^2logn).

Thus, total complexity of Kruskal's Algorithm for computing EMST is O(n^2logn).

Theorem: The EMST of a set of points, S, in 2D is contained within Delaunay triangulation of S. (i.e. EMST is a subset of Delaunay triangulation)

i.e. EMST (S) $\subseteq$ DT (S)

Proof: -



Let us consider an edge (a, b) $\in$ EMST, then we have to show that (a, b) $\in$ DT, for EMST to be the subset of DT. Now, assume the contrary that (a, b) $\in$ EMST and (a, b)$\notin$ DT. Since, (a, b)$\notin$ DT, then no empty circle through a & b i.e. every circle through a & b must have another point inside it. Suppose c is on or inside the circle through a & b (i.e. diameter ab in above fig.)

Then, $|ac| < |ab|$ & $|bc| < |ab|$

Since, (a, b) $\in$ EMST, removal of (a, b) partitions the tree into two trees with a in one part T_a & b in other T_b.

Suppose without loss of generality that c is in Ta. Then removal of (a, b) and addition of (b, c) will make a new tree T' = Ta + bc + Tb. Clearly, this tree has smallest total weight as $|bc| < |ab|$. This means that the EMST using (a, b) that we considered as MST could not have been minimal. This leads to a contradiction. Hence, our supposition (a, b) $\notin$ DT was wrong.

Thus, EMST (S) $\subseteq$ DT (S)