CS 164 & CS 266: Computational Geometry

Lecture 12

Delaunay triangulations and minimum spanning trees

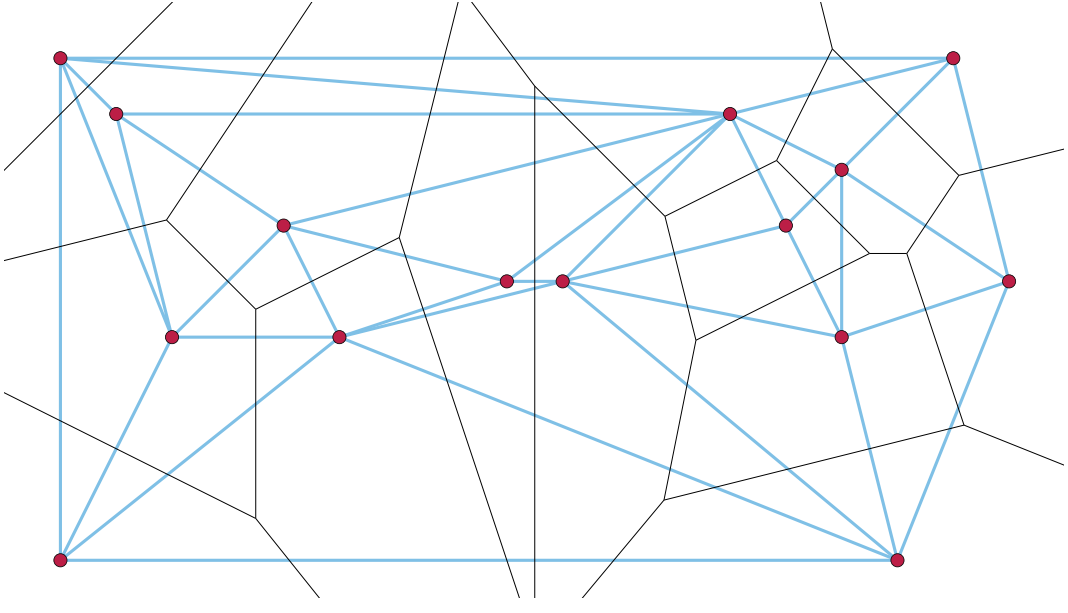
David Eppstein
University of California, Irvine

Fall Quarter, 2025

What is a Delaunay triangulation?

Definition from Voronoi diagram

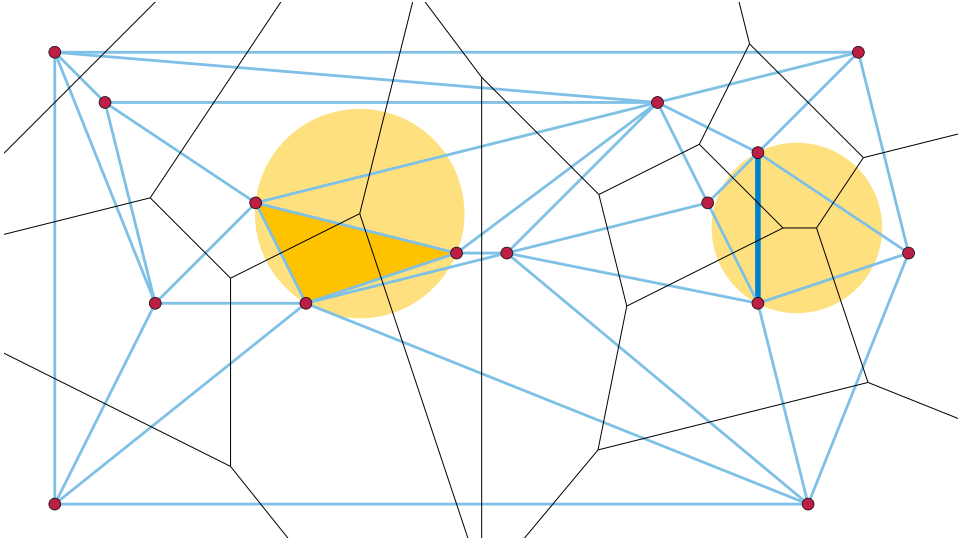
Connect two sites if their Voronoi cells share an edge



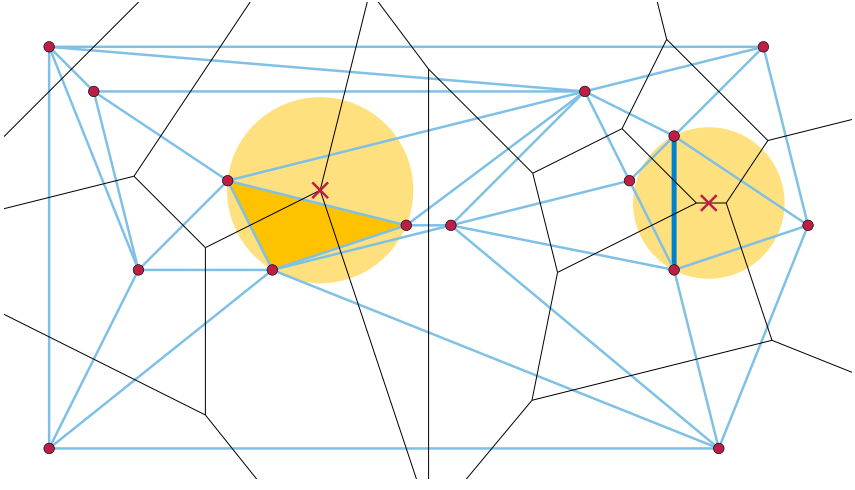
Definition from empty circles

Two vertices form an edge $\iff \exists$ circle only containing them

Three vertices form triangle face $\iff \exists$ circle only containing them



Equivalence of definitions



Center of circle through two sites

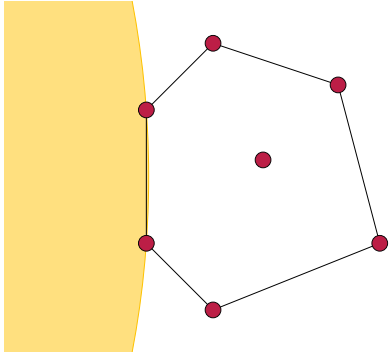
$\Longleftrightarrow$

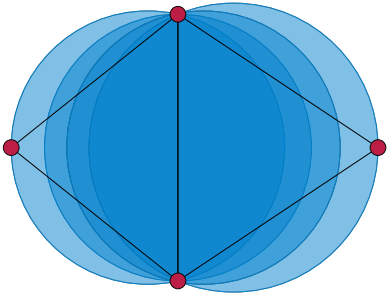
point on Voronoi edge between two cells

Why is this a triangulation?

Edges cannot cross because (to allow endpoints to be in only one circle) their circles would have to cross each other four times, impossible

Outer face is convex hull because its edges all have huge near-halfplane empty circles

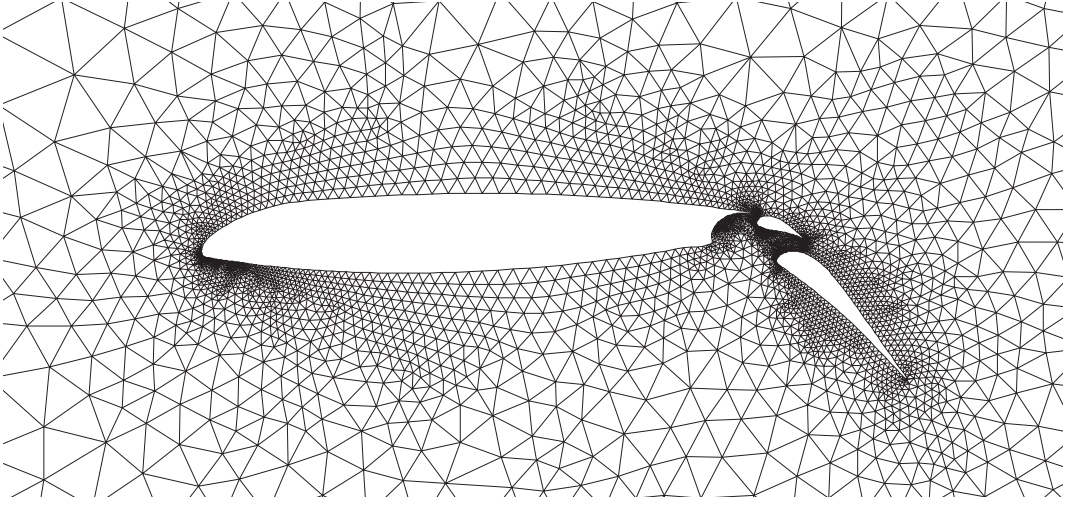




Other edges all belong to two triangles: continuously vary two-circle triangle until hitting 3rd point on either side of the edge

Application in scientific computing

Commonly used in finite element mesh generation to generate a mesh after some other method (e.g. quadtrees, week 3) has been used to place points for vertices

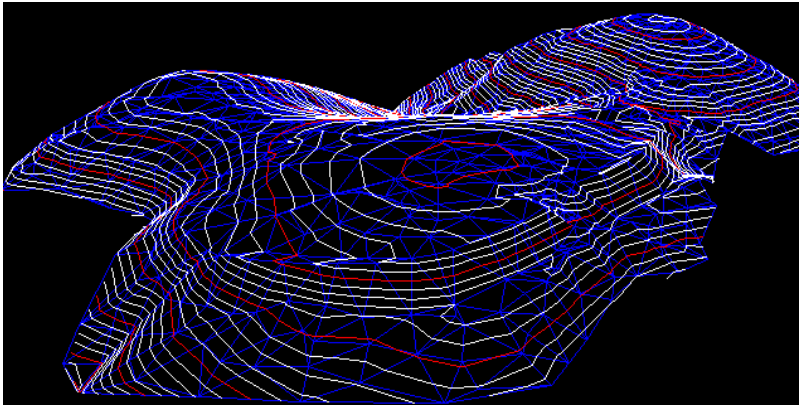


Application in geographic information systems

Triangulation of points with elevations $\Rightarrow$ surface in 3d

Given irregularly placed measurements of ground elevation, connect to form 3d model of ground surface

Called a "triangulated irregular net"



Application in face recognition

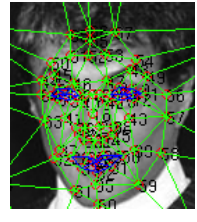
"DeepFace": used by Facebook to recognize people in photos from 2014 to 2021 (stopped for legal reasons, not technical problems)

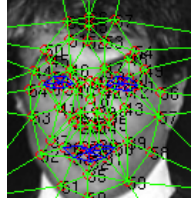
Uses six points ($2 \times$ eyes, nose, $3 \times$ mouth) to fit 3d generic model

Map 67 "fiducial marks" on 3d model back to 2d image; Delaunay triangulate

Linearly map each triangle to warp to symmetric "frontal" appearance

Result is passed to a deep neural net





[Taigman et al. 2014]

Optimality

Among all triangulations of a given set of points, Delaunay is optimal for many measures of triangle quality:

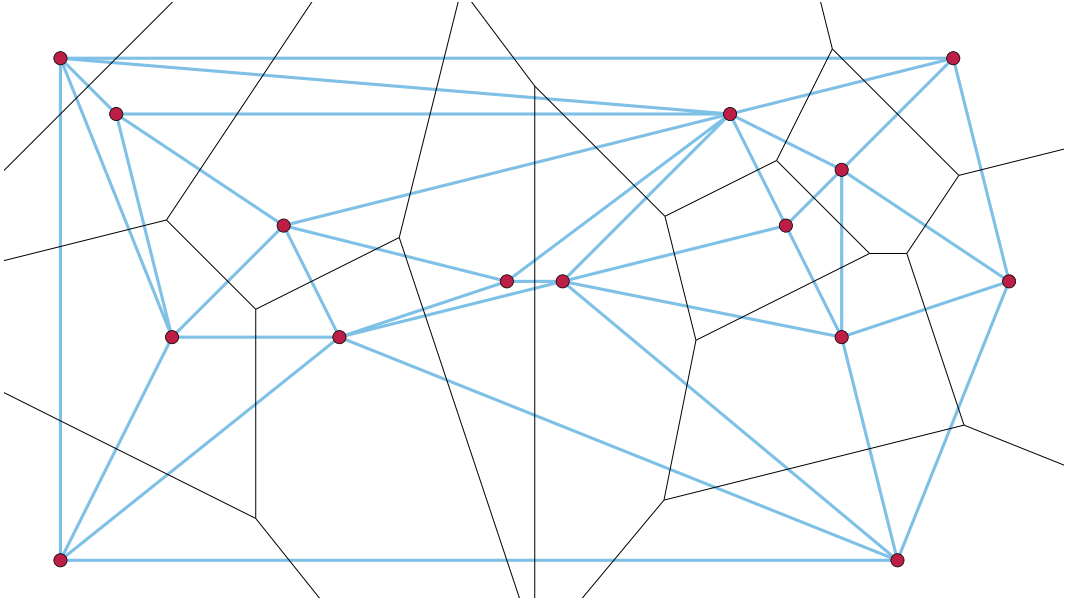
- It avoids sharp angles: it makes the smallest angle as large as possible
- It finds small triangles: it makes the smallest circle around any triangle as small as possible (either for the circle through three vertices, or the smallest circle containing the triangle)
- For 3d surfaces from ground elevation, it finds a surface that avoids steep slopes: define "energy" of a triangle = squared gradient times projected area, then it minimizes the sum of energy of all triangles

Proof idea: Later this lecture

Delaunay triangulation algorithms

Algorithm by Voronoi dual

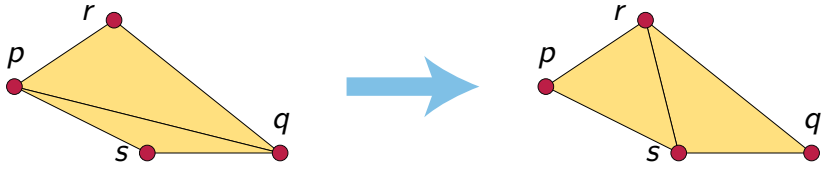
Construct Voronoi diagram using Fortune's algorithm



Algorithm by flipping

Start with any triangulation (for instance, the plane sweep triangulation we used for visibility in Week 3)

Bad = collection of edges pq such that, for its two triangles pqr and pqs, the Delaunay triangulation of quadrilateral $prqs$ uses the other diagonal, rs



while Bad is not empty:

- Find an edge pq in Bad, and remove it from Bad
- Replace the two triangles pqr and pqs by prs and qrs
- Check whether any of the four edges of quadrilateral $prqs$ become bad or stop being bad

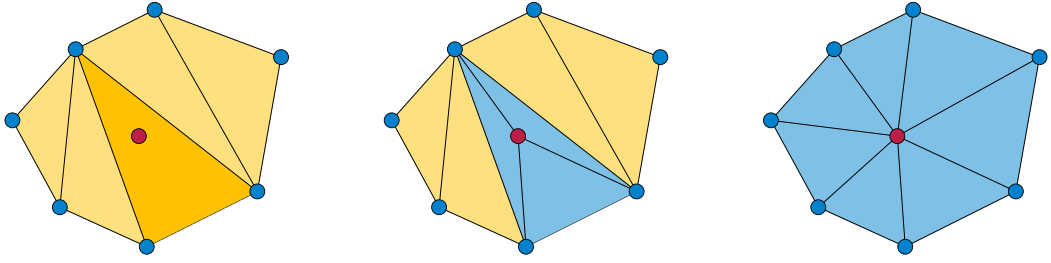
Correctness and analysis: Not yet

Algorithm by random incremental flipping

Add three artificial points whose convex hull surrounds whole input; start with one-triangle triangulation

For each given point p, in random order:

- Use a history DAG to find the triangle containing p
- Subdivide it into three triangles meeting at p
- Repeatedly flip non-Delaunay quadrilaterals involving p remembering history of all flips in history DAG



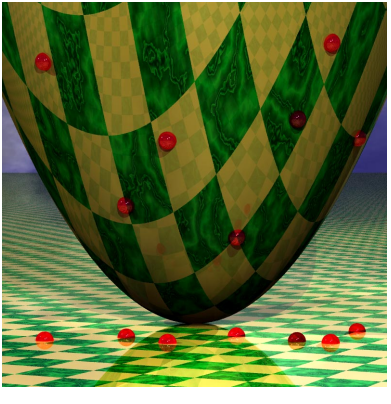
Correctness and analysis: Not yet

Algorithm by 3d convex hulls

"Lift" 2d points (x, y) to 3d points $(x, y, x^2 + y^2)$

Compute lower convex hull

Project back down to 2d



Time $O(n \log n)$ (Week 4)

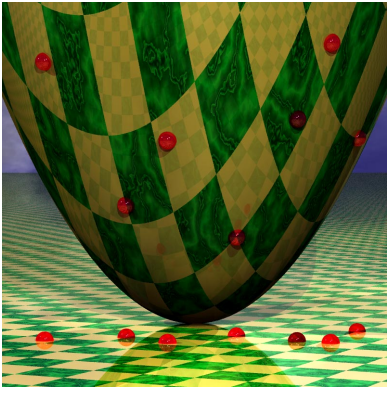
Correctness: Next slide

Lifting and empty circles

"Lift" 2d points (x, y) to 3d points $(x, y, x^2 + y^2)$

Circles in the plane lift to intersections of plane in 3d with the surface $z = x^2 + y^2$

Circle is empty $\iff$ halfspace below 3d plane is empty



Therefore, lower hull faces (sets of points touched by boundaries of empty lower halfspaces $\iff$ Delaunay edges and triangles (sets of points touched by empty circles

Correctness of flipping

Flipping a non-Delaunay quadrilateral to its Delaunay triangulation corresponds, in the 3d hull, to gluing a tetrahedron below a non-convex edge of a triangulated surface, making the surface convex downwards at the new edge

Once we glue a tetrahedron below a non-convex edge, the triangulated surface only moves downward, never back up $\Rightarrow$ we will never see that edge again $\Rightarrow$ there can be at most $\binom{n}{2}$ flips

If the triangulated surface is not already the lower convex hull, it must have a non-convex edge, onto which we can glue a tetrahedron

Therefore, the flipping algorithms cannot run out of flips until they reach the Delaunay triangulation

Analysis of flipping algorithms

Basic flipping: $O(1)$ time per flip, at most $\binom{n}{2}$ flips $\Rightarrow$ $O(n^2)$ time

Randomized incremental: When using the history DAG for point location of a point p, probability that after inserting i random points, the triangle containing p has the last point as one of its vertices $= 3/(i+3)$ $\Rightarrow$ sum over i is a harmonic series $\Rightarrow$ total expected time to locate p is $O(\log n)$

Probability that after inserting i random points, a given triangle in the current Delaunay triangulation was just added $= 3/(i+3)$ $\Rightarrow$ expected number of new triangles is $O(1)$ $\Rightarrow$ expected number of flips per added point is $O(1)$

$O(n \log n)$ total time for point location, $O(n)$ for flipping
$O(n)$ triangles and flips in history DAG

$O(n \log n)$ expected time, $O(n)$ expected space

Optimality revisited

Among all triangulations of a given set of point, Delaunay is optimal for many measures of triangle quality:

- It makes the smallest angle as large as possible
- It makes the smallest circle around any triangle as small as possible (either for the circle through three vertices, or the smallest circle containing the triangle)
- For 3d surfaces from ground elevation, define "energy" of a triangle = squared gradient times projected area, then it minimizes the sum of energy of all triangles

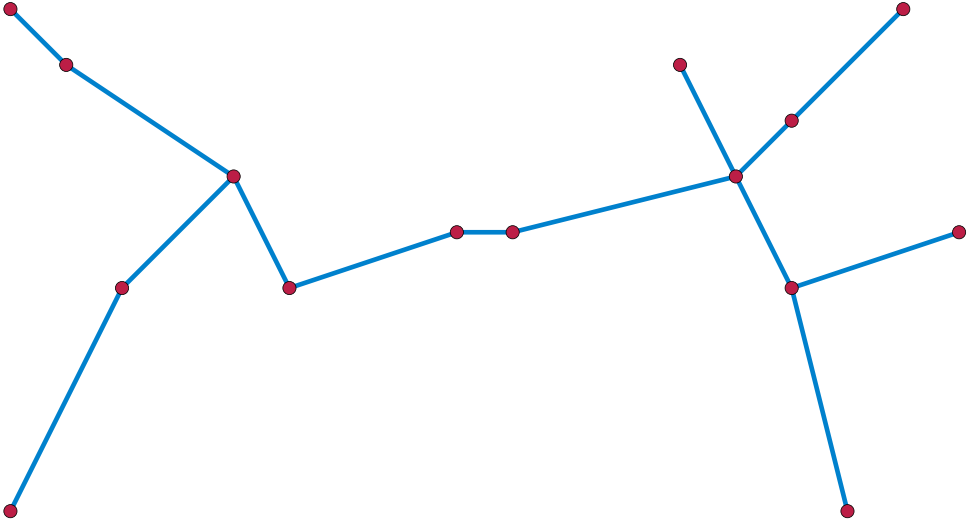
Proof idea: Show that Delaunay flips improve each of these

Flipping cannot get stuck in any local optima $\Rightarrow$ always finds optimal triangulation

Minimum spanning trees

Euclidean minimum spanning tree

Connect given points by a tree of minimum total edge length



Applications of minimum spanning tree

Original application: making physical connections between geographic locations (power grid) with low construction cost

Clustering: delete longest edge $\Rightarrow$ two clusters as far from each other as possible

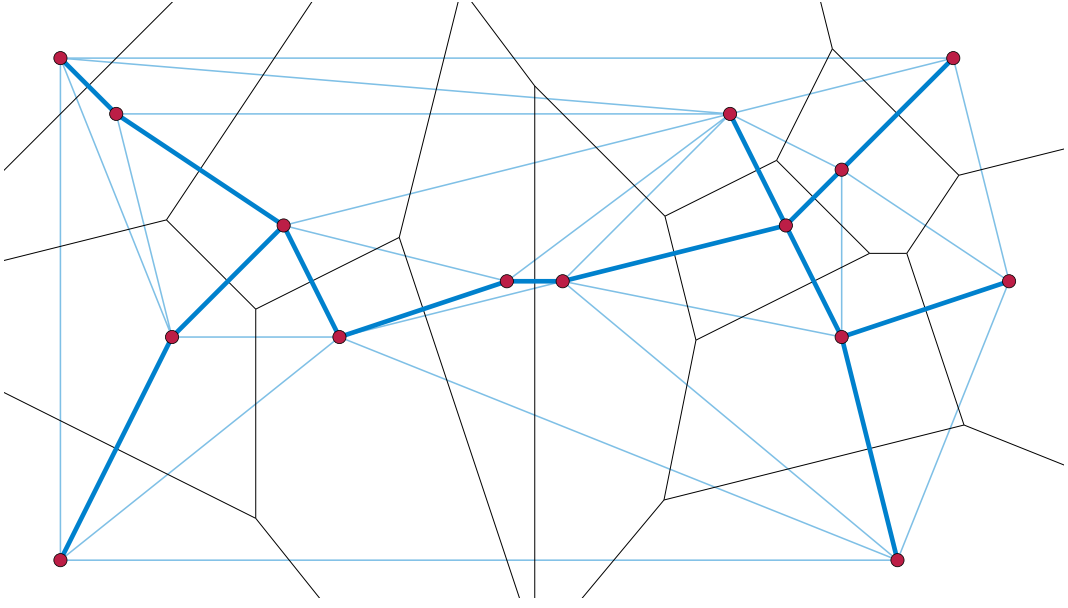
Generating a one-dimensional approximation to the shape of a cloud of points

Approximating traveling salesperson tour
(tree traversal order gives tour of length $\leq 2 \times$ optimal length)

Speed up nearest neighbor classification by removing points when it cannot change any classification [Eppstein 2022]

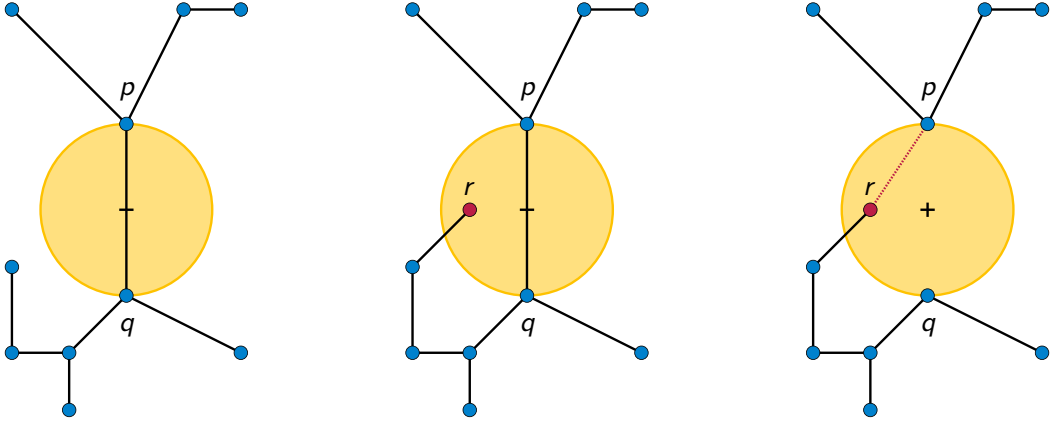
Minimum spanning tree property

Every edge of the minimum spanning tree is in the DT



The empty circle for an MST edge

For an edge pq, consider the circle with pq as its diameter



If circle contains a point r, we get a better tree by removing pq and reconnecting using pr or qr (both shorter than pq)

So it's an empty circle $\Rightarrow$ pq is in the Delaunay triangulation

To construct MST in time $O(n \log n)$:

- Find the Delaunay triangulation, a graph with $m = O(n)$ edges
- Use any algorithm for finding minimum spanning trees in graphs in time $O(m \log n)$ or faster (for example, the Prim–Dijkstra–Jarník algorithm, Borůvka's algorithm, or Kruskal's algorithm)

Because the Delaunay triangulation is a planar graph (it has no crossings), Borůvka's algorithm can be made to run in $O(n)$, but this doesn't help because constructing the Delaunay triangulation still takes $O(n \log n)$ time

References and image credits

David Eppstein. Finding relevant points for nearest-neighbor classification. In *Proc. SIAM Symp. Simplicity in Algorithms (SOSA 2022)*, pages 68–78, 2022. doi: 10.1137/1.9781611977066.6.

Robert Kropf. Digitales Geländemodell (die Höhensiedlung und Wallanlage Königsberg bei Tieschen, Steiermark, Österreich). Licensed under the Creative Commons Attribution-Share Alike 3.0 Germany license, March 1999. URL `https://commons.wikimedia.org/wiki/File:Digitales_Gel%C3%A4ndemodell.png`.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23–28, 2014*, pages 1701–1708. IEEE Computer Society, 2014. doi: 10.1109/CVPR.2014.220.