Wolfgang Mulzer
Institut für Informatik

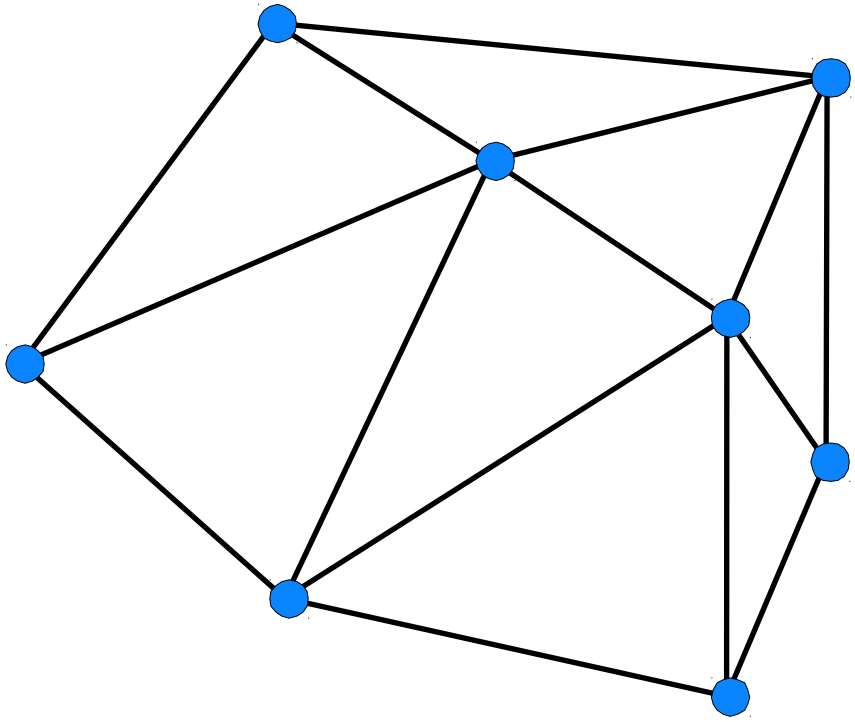
Freie Universität Berlin

Planar Delaunay Triangulations and Proximity Structures

Proximity Structures

Given: a set P of n points in the plane

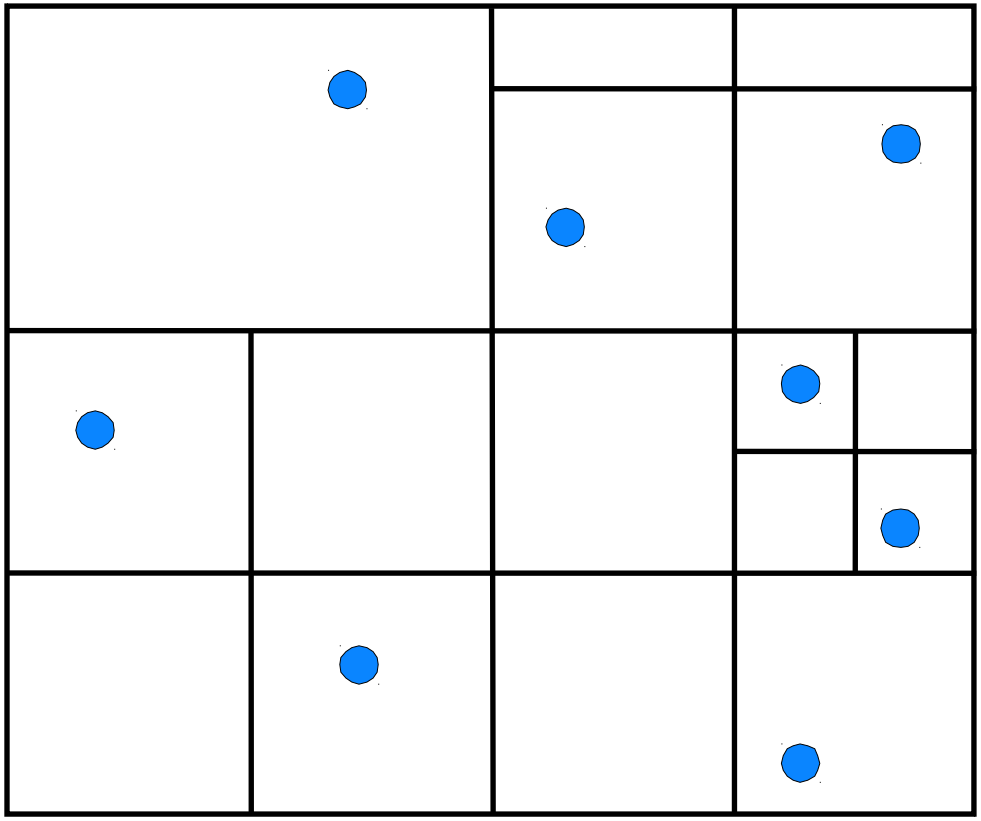
proximity structure: a structure that "encodes useful information about the local relationships of the points in P"



Proximity Structures

Given: a set P of n points in the plane

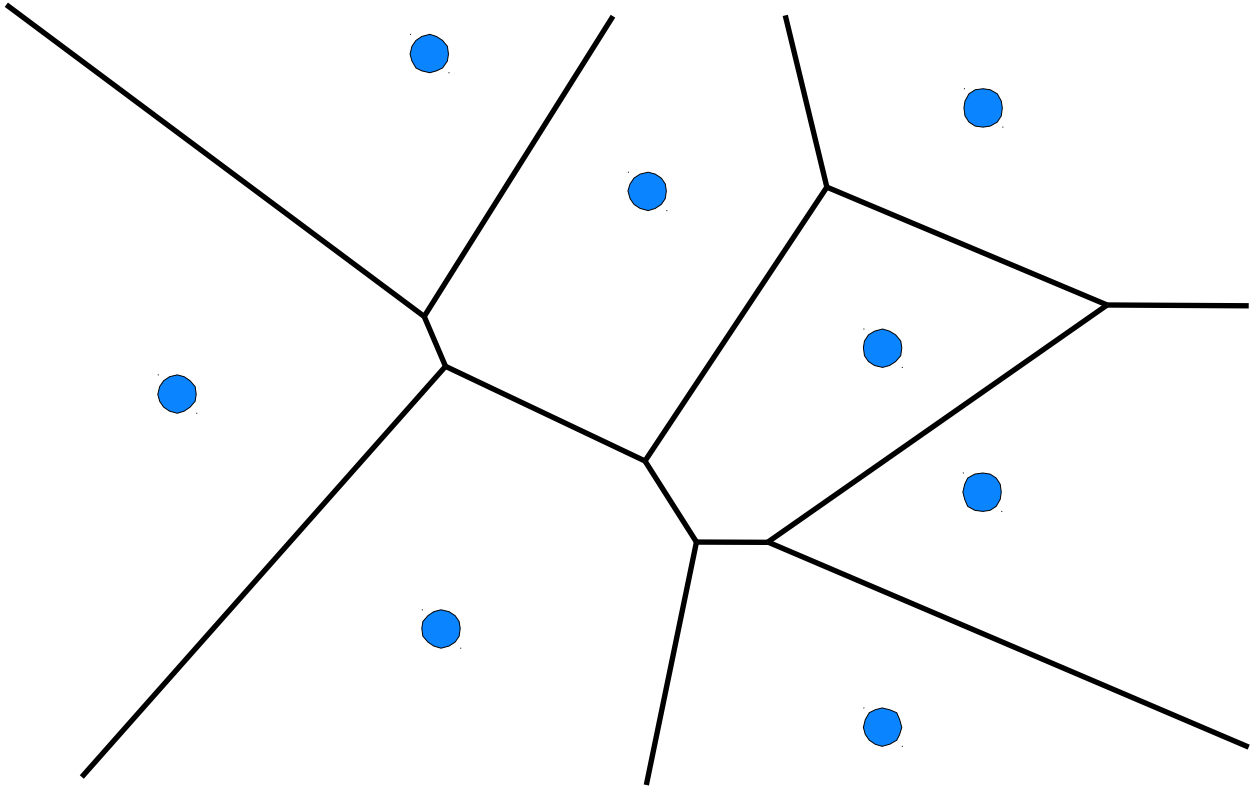
proximity structure: a structure that "encodes useful information about the local relationships of the points in P"



Proximity Structures

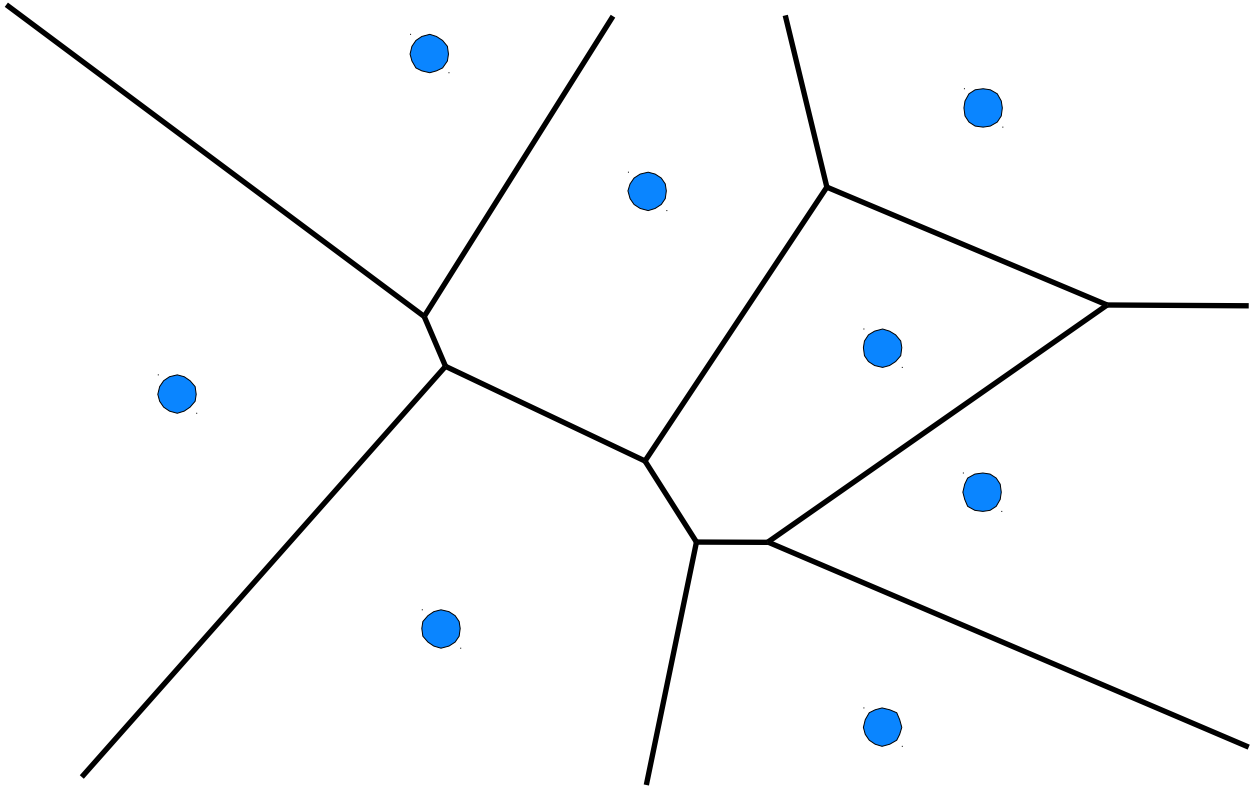
Given: a set P of n points in the plane

proximity structure: a structure that "encodes useful information about the local relationships of the points in P"



Proximity Structures

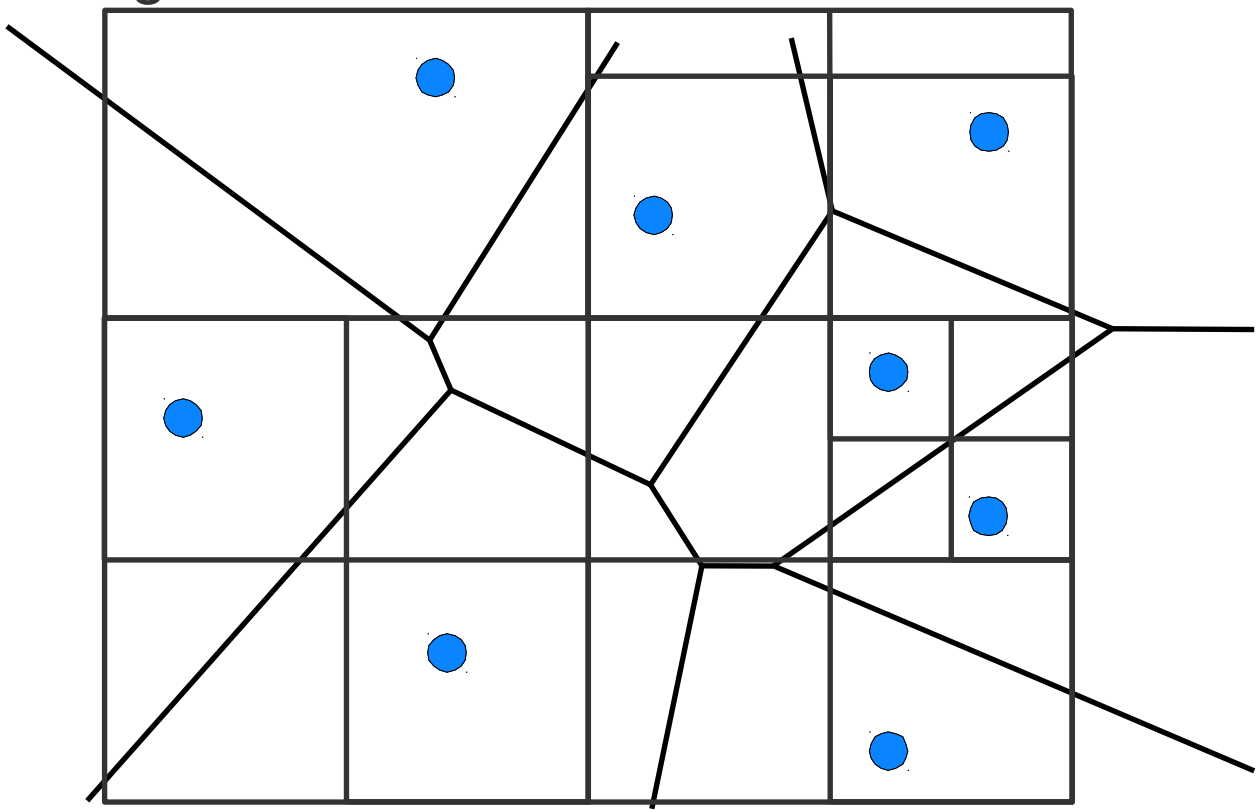
Reduction from sorting
$\rightarrow$ usually need $\Omega(n \log n)$ to build a proximity structure



Proximity Structures

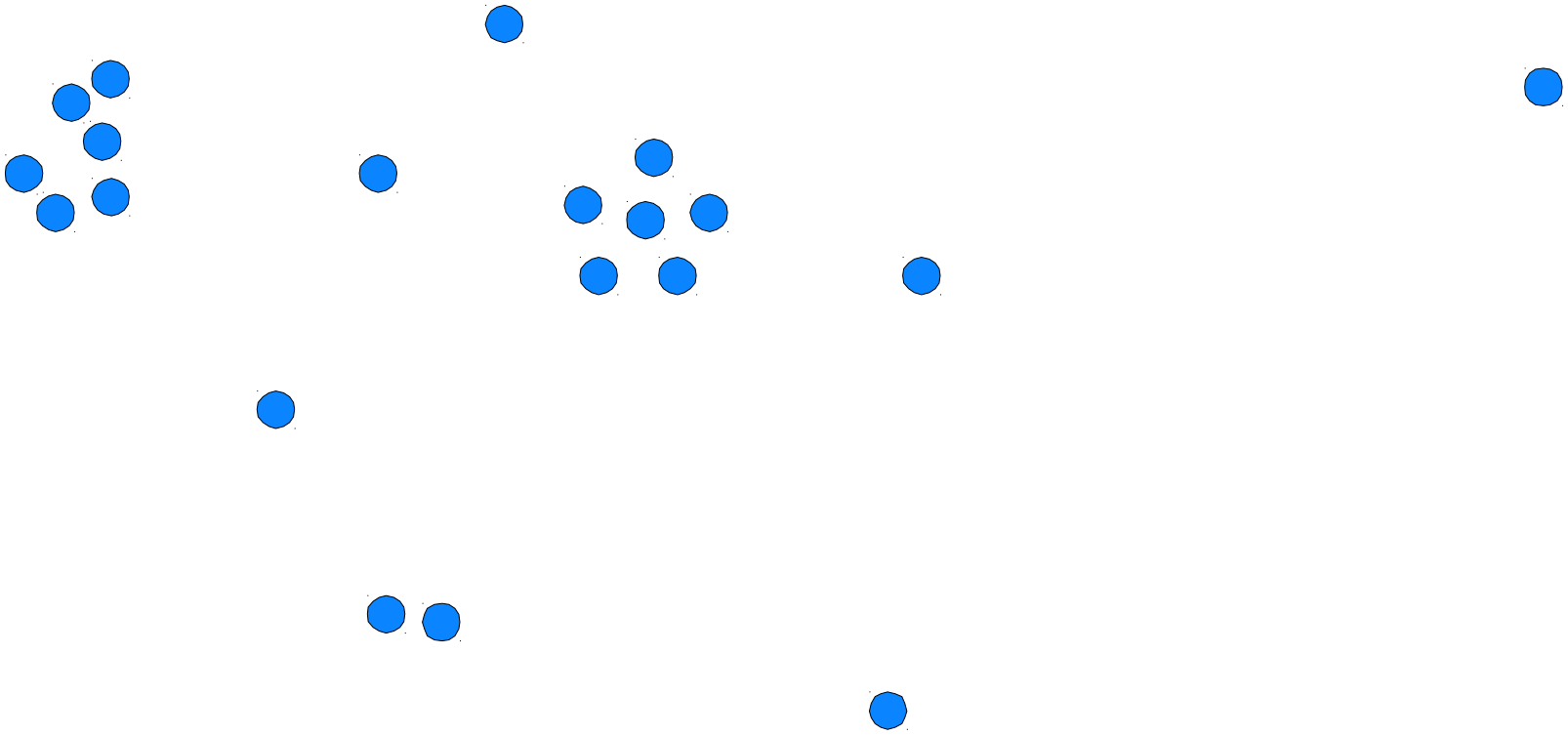
But: shouldn't one proximity structure suffice to construct another proximity structure faster?

Voronoi diagram → Quadtree



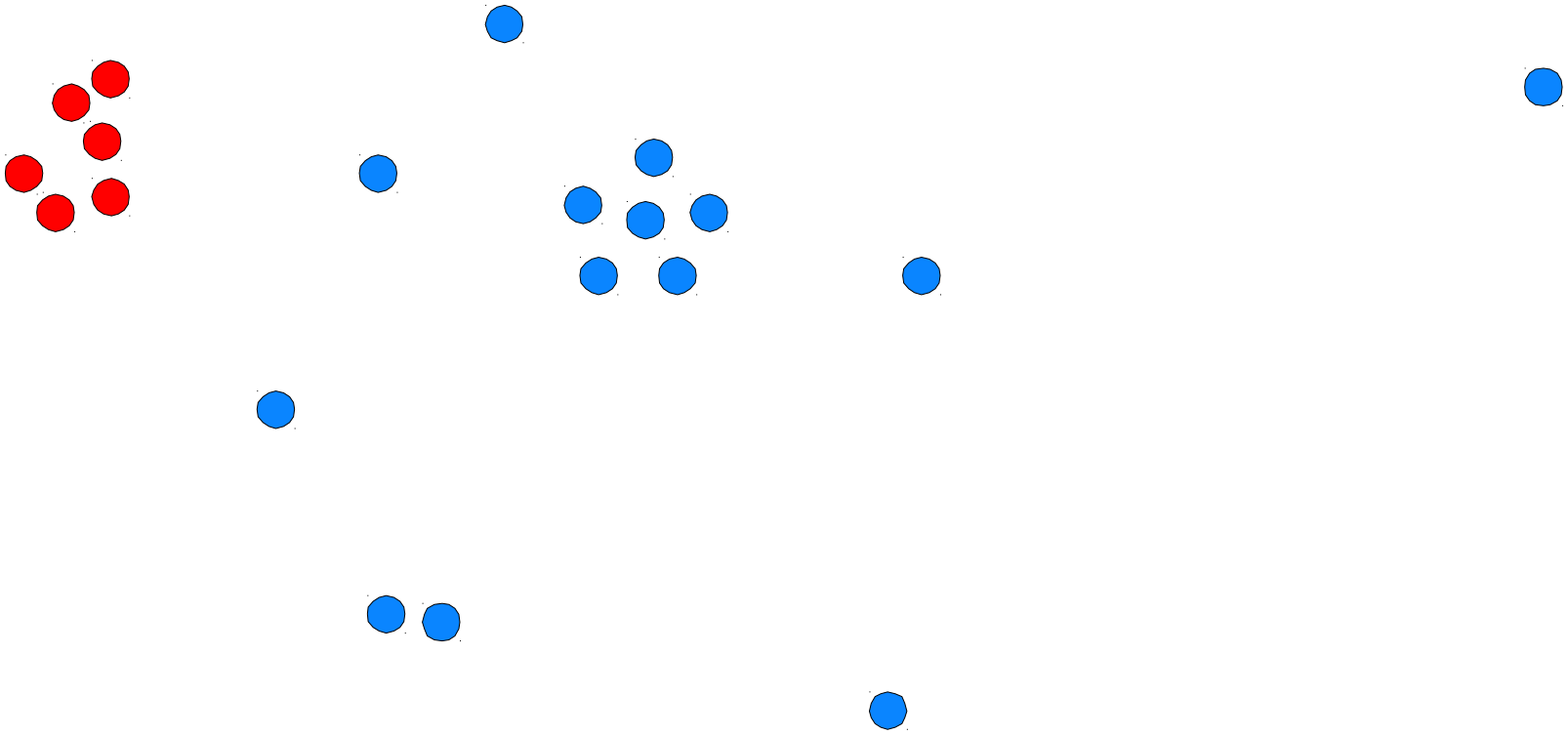
Proximity Structures

Point sets may exhibit strange behaviors, so this is not always easy.



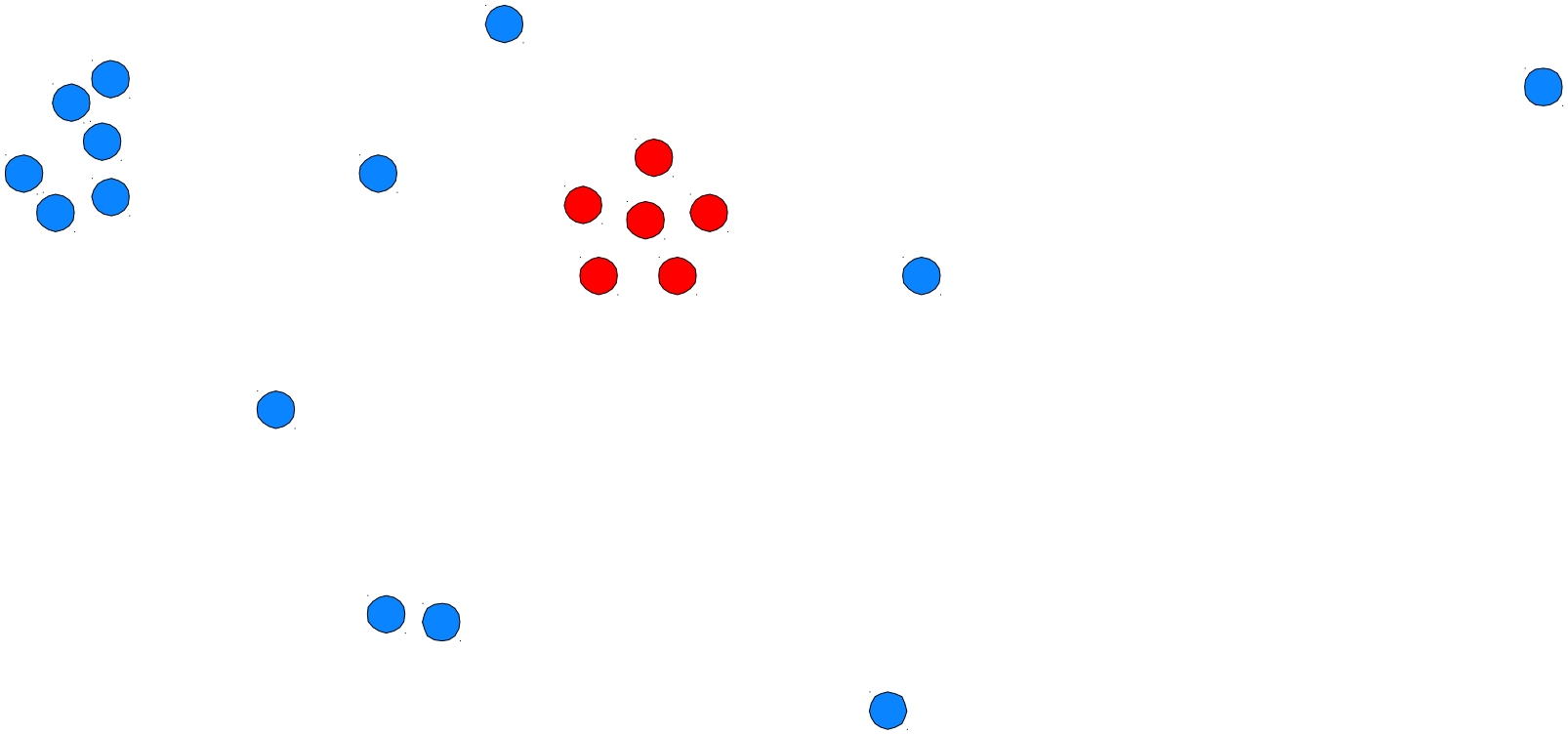
Proximity Structures

There might be clusters…



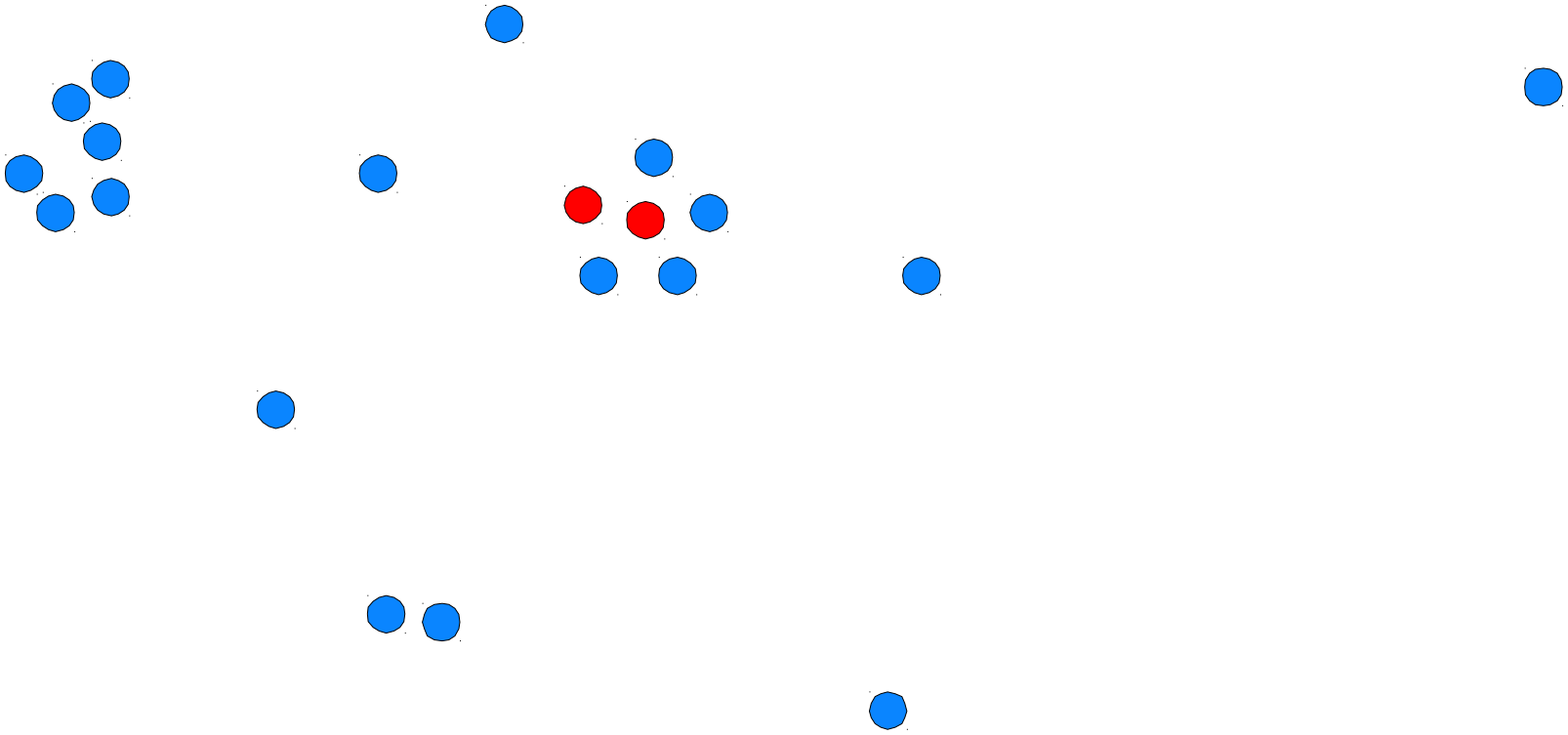
Proximity Structures

…high degrees…



Proximity Structures

or large spread.



Overview

[Dirichlet, 1850]

[Delaunay, 1934]

[Gabriel, 1969]

[Clarkson, 1983]

[Preparata Shamos 1985]

[Bern Eppstein Gilbert 1990]

[Chazelle 1991]

[Matsui 1995]

[Callahan Kosaraju 1995]

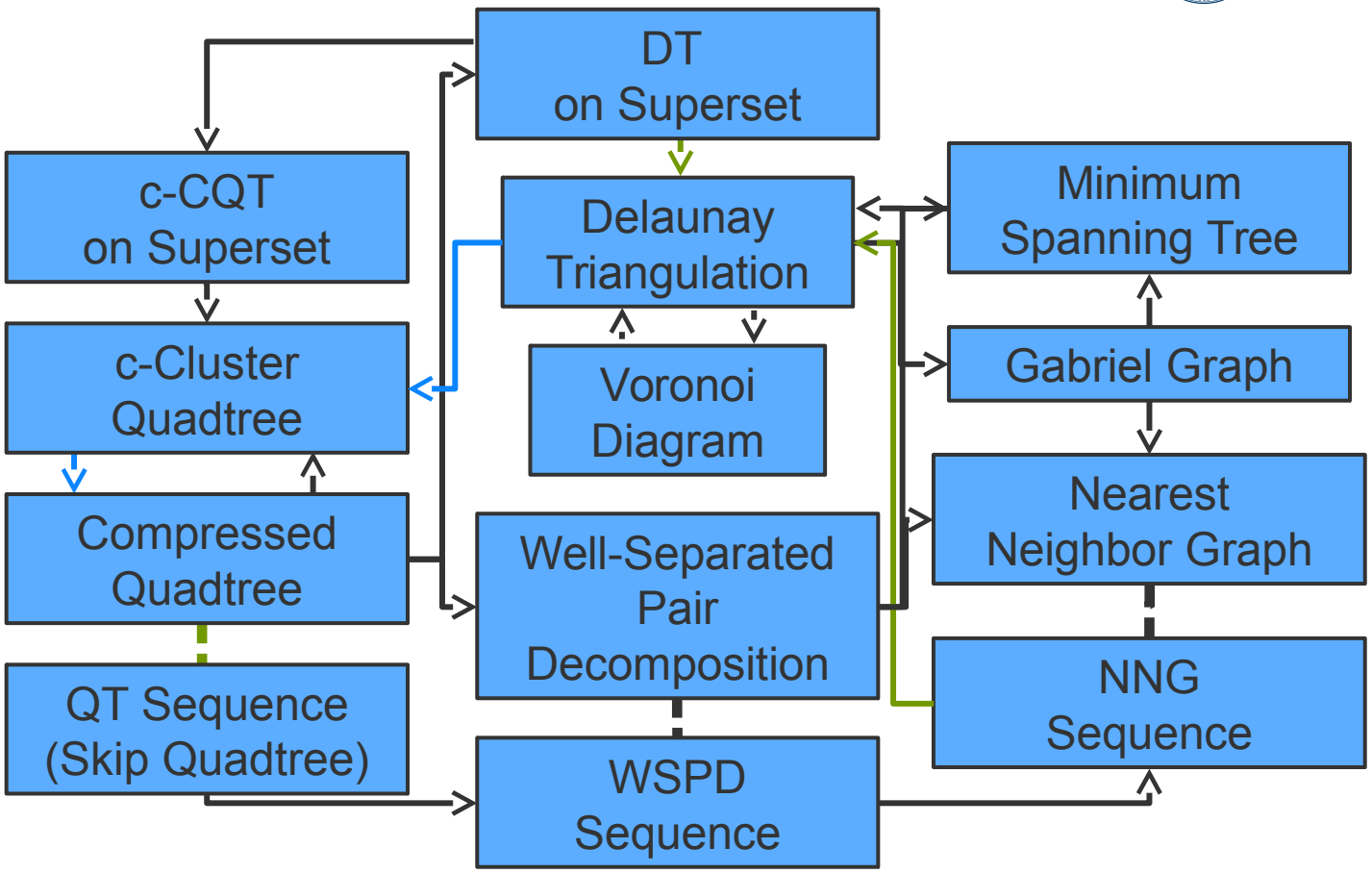
[Chin Wang1998]

[Krznaric Levcopoulos1998]

[Chazelle D H M S T 2002]

[Eppstein Goodrich Sun 2005]

[Buchin M 2009] [Löffler M 2012]



Linear Time (deterministic)

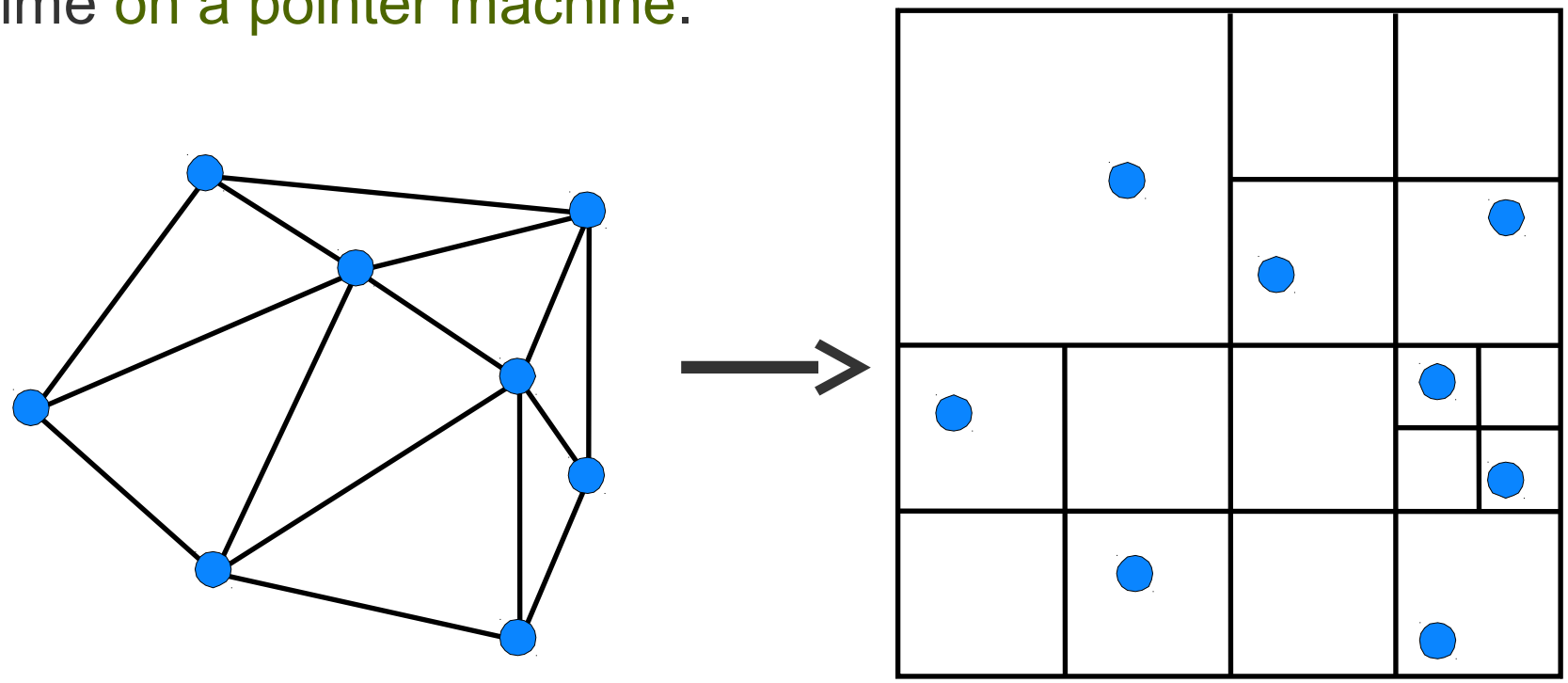
Linear Time (randomized)

Linear Time (w/ floor function)

Our Results

Two algorithms:

Given: Delaunay triangulation for P
Can find: compressed quadtree for P in linear deterministic time on a pointer machine.



Our Results

Two algorithms:

Given: Delaunay triangulation for P
Can find: compressed quadtree for P in linear deterministic time on a pointer machine.

Previous result by Levcopolous and Krznaric [1998] uses bit tricks and bucketing.

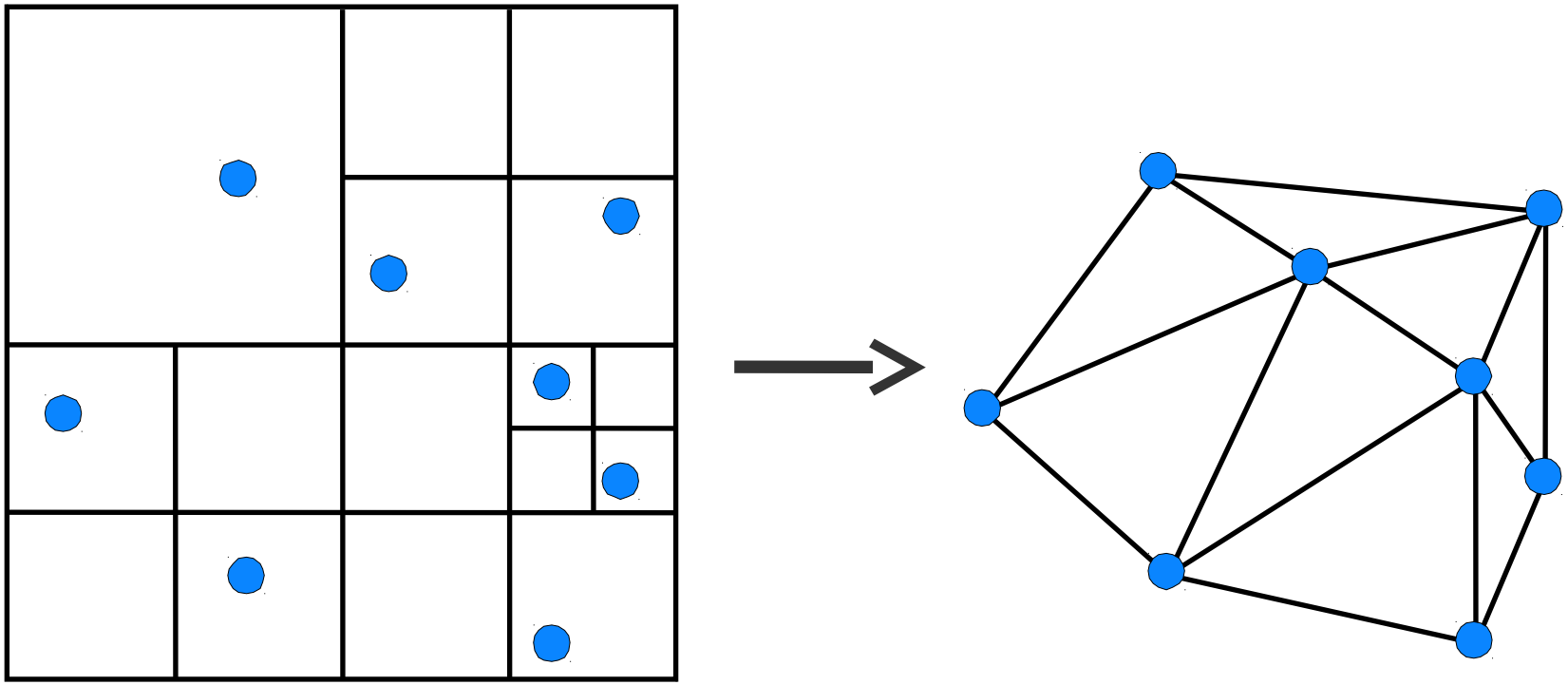
Our result follows by carefully adapting their algorithm to avoid these features.

Won't give any more details here.

Our Results

Two algorithms:

Given: compressed quadtree for P
Can find: a Delaunay triangulation for P in linear time on a pointer machine.



Our Results

Two algorithms:

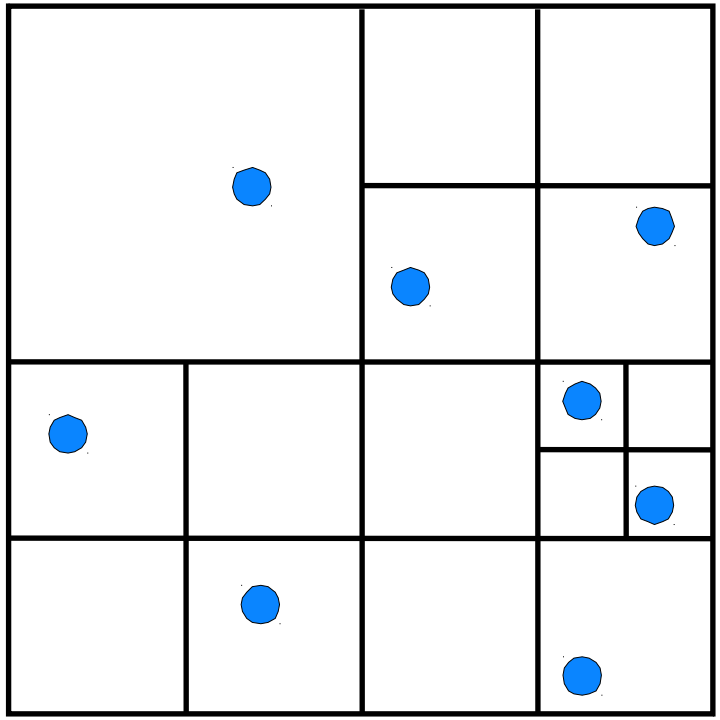
Given: compressed quadtree for P
Can find: a Delaunay triangulation for P in linear time on a pointer machine.

Randomized algorithm: relatively simple, incremental construction of the Delaunay triangulation (w. K. Buchin).

Deterministic algorithm: different approach and several new ideas (w. M. Löffler).

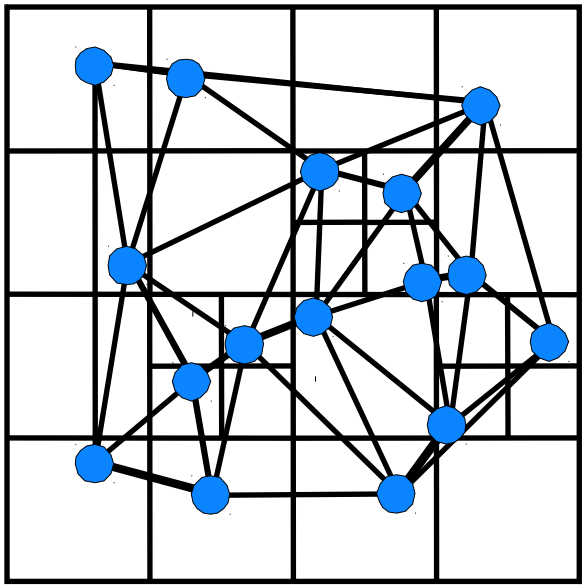
Compressed Quadtrees

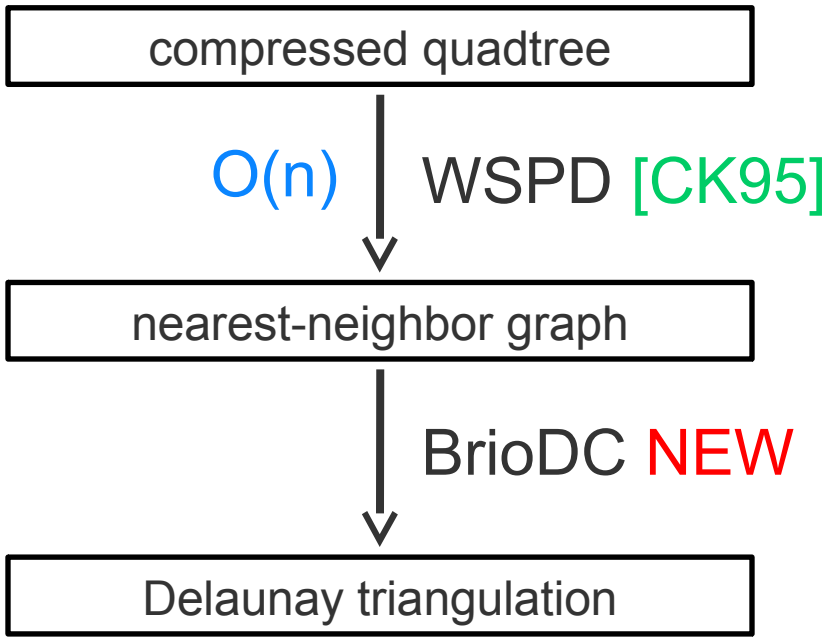
Quadtree: hierarchical subdivision of a bounding square for P into axis parallel boxes that separate P.



Can be compressed if P is very clustered.

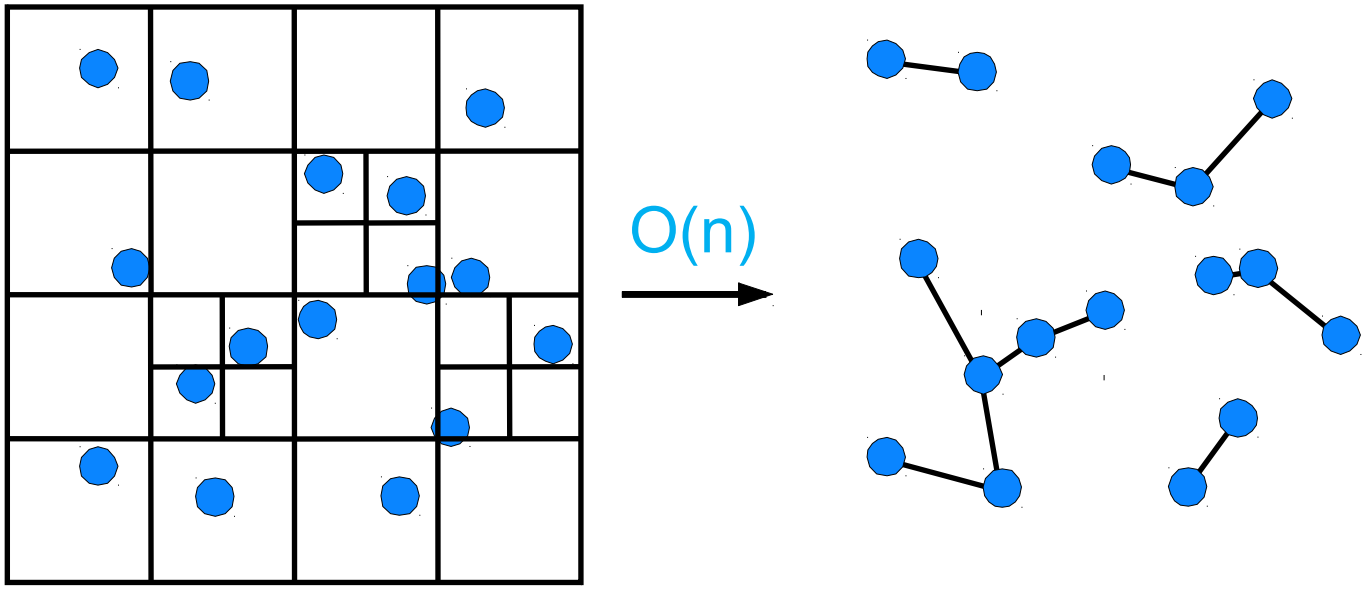
Randomized Algorithm





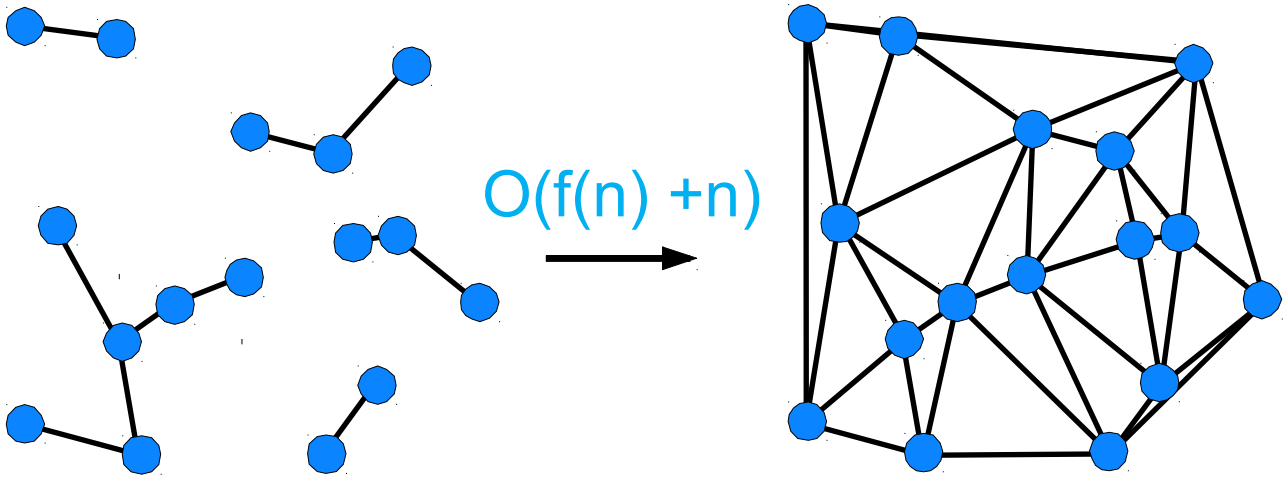
Quadtrees and Nearest-Neighbor Graphs

Theorem [CK95]: Given a compressed quadtree for P, we can find its nearest-neighbor graph in time O(n) using the well-separated pair decomposition for P.



Nearest-Neighbor Graphs and DTs

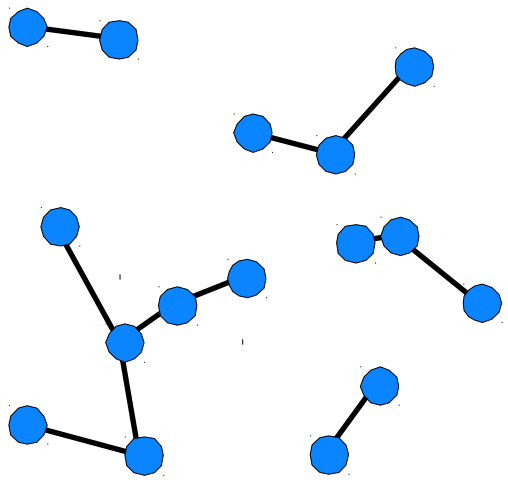
Theorem: If the nearest-neighbor graph for any Q can be found in time f(|Q|), with f(|Q|)/|Q| nondecreasing, the Delaunay triangulation of P can be found in expected time O(f(n)+n).



Nearest-Neighbor Graphs and DTs

Theorem: If the nearest-neighbor graph for any Q can be found in time f(|Q|), with f(|Q|)/|Q| nondecreasing, the Delaunay triangulation of P can be found in expected time O(f(n)+n).

Proof: We use a randomized incremental construction with biased insertion order and dependent sampling.



Given P.

Find NNG(P).

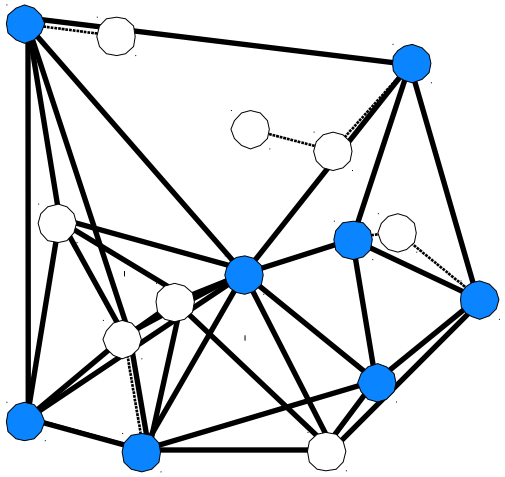
Pick an edge in each component.

Sample one point from each edge, sample the rest independently.

Nearest-Neighbor Graphs and DTs

Theorem: If the nearest-neighbor graph for any Q can be found in time f(|Q|), with f(|Q|)/|Q| nondecreasing, the Delaunay triangulation of P can be found in expected time O(f(n)+n).

Proof: We use a randomized incremental construction with biased insertion order and dependent sampling.



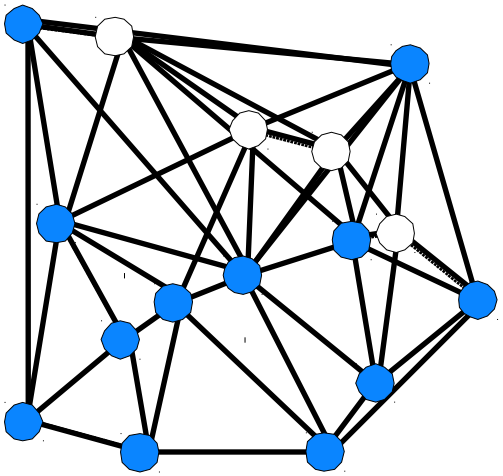
Recurse on the sample.

Insert the remaining points: walk along edges of NNG(P) and insert the points along the way.

Nearest-Neighbor Graphs and DTs

Theorem: If the nearest-neighbor graph for any Q can be found in time f(|Q|), with f(|Q|)/|Q| nondecreasing, the Delaunay triangulation of P can be found in expected time O(f(n)+n).

Proof: We use a randomized incremental construction with biased insertion order and dependent sampling.



Recurse on the sample.

Insert the remaining points: walk along edges of NNG(P) and insert the points along the way.

Analysis

Lemma: The time to insert the remaining points is $O(|P|)$.

The sample size decreases geometrically, so the lemma gives total $O(f(n)+n)$.

Proof idea for the lemma:

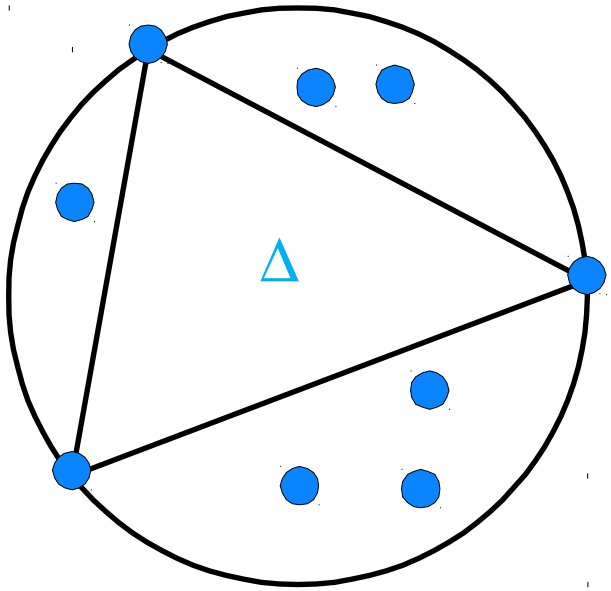
$NNG(P) \subseteq DT(P)$, so all triangles traversed during an insertion step will be destroyed.

Hence, it suffices to count the number of active triangles in the insertion phase (structural change).

Analysis

Consider a triangle Δ.

Δ can be active during the insertion phase only if the sample contains no point inside its circumcircle.

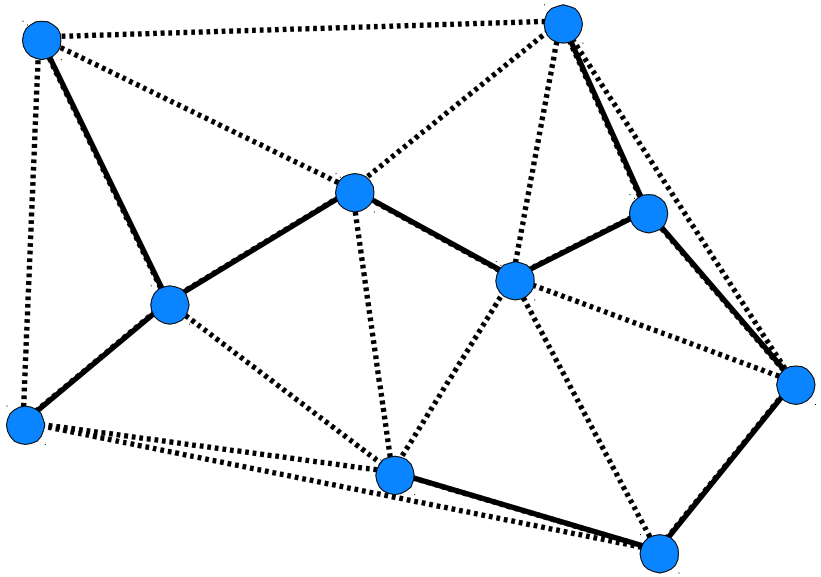


If Δ's circumcircle contains s points, the probability for this event is $\leq 1/2s$.

There are $O(ns2)$ empty triangles with $\leq s$ points in their circumcircle [CS88], so the expected number of active triangles is $O(n\sum ss2/2s) = O(n)$.

Deterministic Algorithm

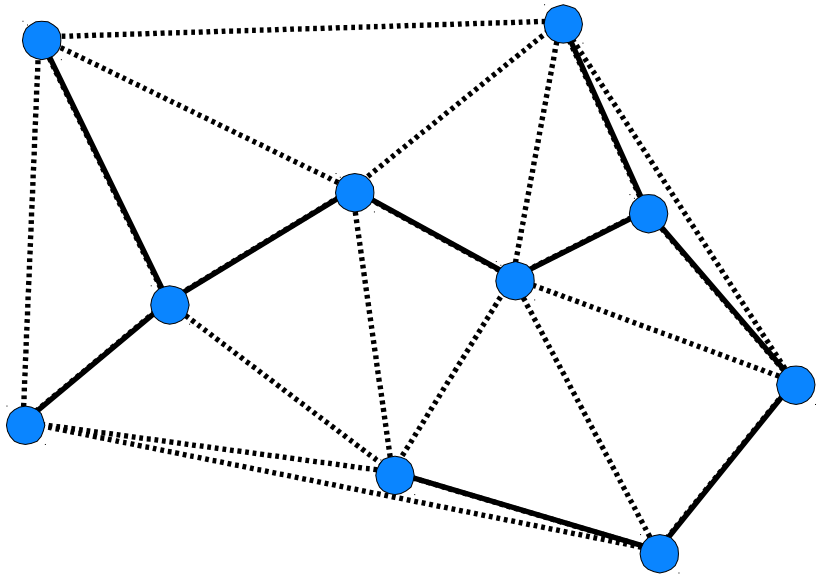
Given a quadtree for P, we want the Delaunay triangulation for P. Actually, we compute the Euclidean minimum spanning tree for P: the shortest tree with vertex set P.



The EMST is a subgraph of the Delaunay triangulation.

Deterministic Algorithm

Given a quadtree for P, we want the Delaunay triangulation for P. Actually, we compute the Euclidean minimum spanning tree for P: the shortest tree with vertex set P.



Given the EMST, we can find the DT in linear deterministic time on a pointer machine [ChinWang98].

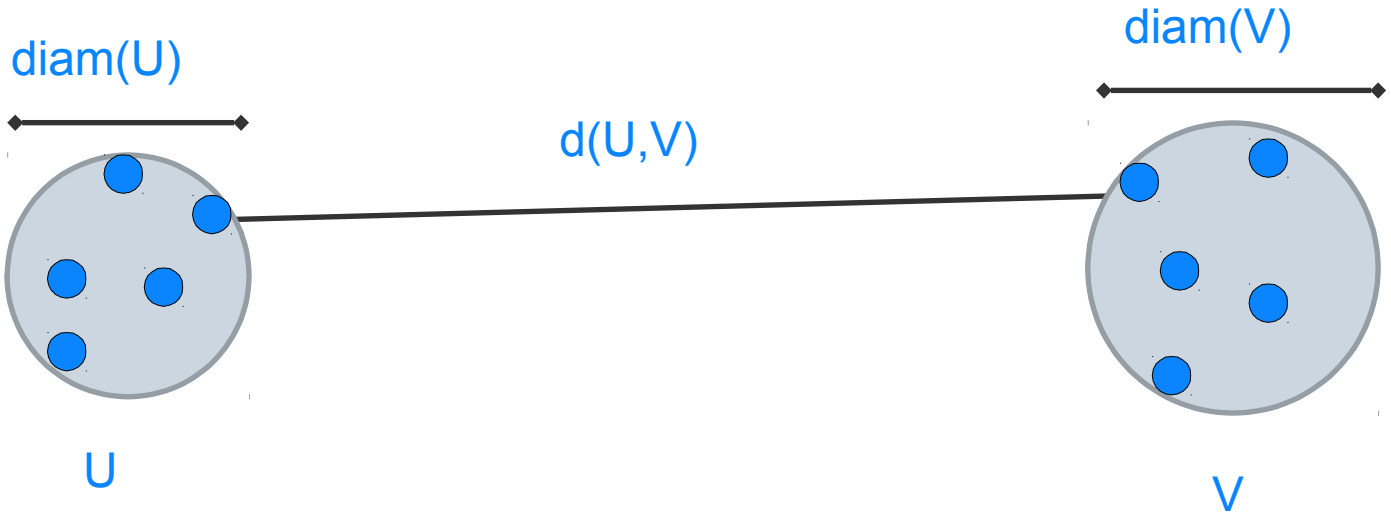
The Final Link

We need a structure that connects quadtrees and Euclidean MSTs – the well-separated pair decomposition (WSPD) [CK95].

A WSPD offers a way to approximate the $\Theta(n^2)$ distances in an n-point set compactly.

WSPD - Definition

Two point sets U and V are ε-well separated, if
$\max(\text{diam}(U), \text{diam}(V) \leq \varepsilon d(U,V)$.



If we represent U by an arbitrary point p ε U, and V by an arbitrary q ε V, we lose only a factor 1±ε in the distance.

WSPD - Definition

Given an n-point set P, an ε-well separated pair decomposition for P is a set of pairs {(U1,V1), (U2, V2), …, (Um, Vm)} so that

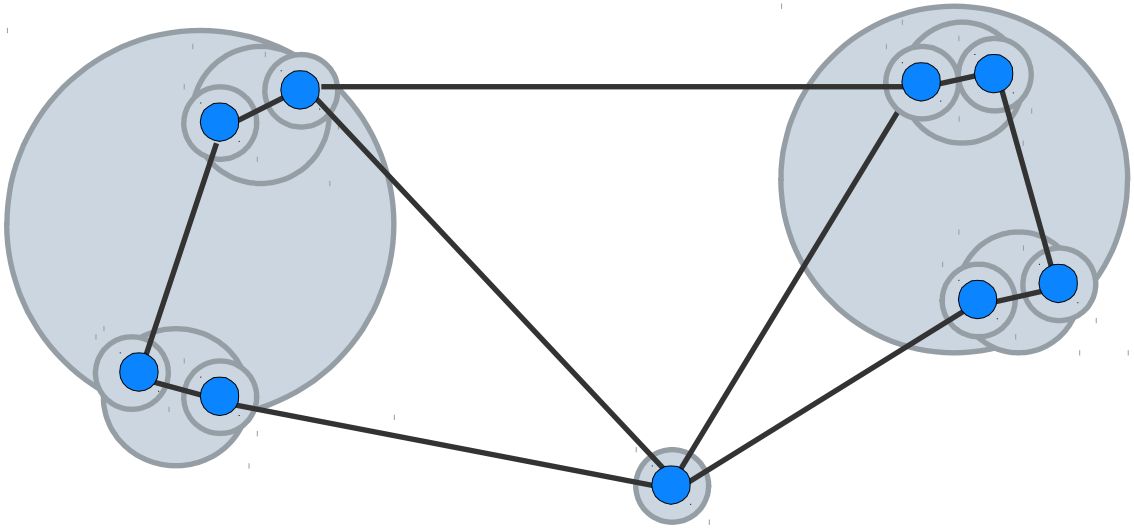
1. For every i, we have Ui, Vi $\subseteq$ P and Ui, Vi are ε-well separated.

2. For every distinct p, q ε P, there is exactly one pair (Ui,Vi) with p ε Ui and q ε Vi or vice versa.

Given a compressed quadtree for P, an ε-WSPD with O(n) pairs can be found in linear deterministic time on a pointer machine [CK95].
The ε-WSPD is represented as pointers to nodes in the quadtree.

WSPD and EMST

Lemma: Let {(U1,V1), (U2, V2), …, (Um, Vm)} be an ε-WSPD for P, and let G be the graph on P that for each i has an edge between the closest pair between Ui and Vi. Then G contains the EMST of P.

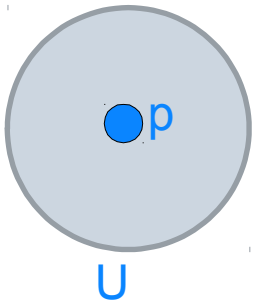


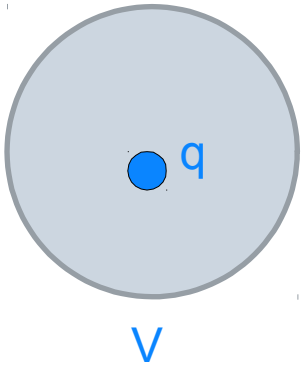
Note: G has only m = O(n) edges.
The lemma is well known and used often in the literature.

WSPD and EMST

Lemma: Let {(U1,V1), (U2, V2), …, (Um, Vm)} be an ε-WSPD for P, and let G be the graph on P that for each i has an edge between the closest pair between Ui and Vi. Then G contains the EMST of P.

Proof: pq: edge of the EMST for P,
(U,V): pair of the WSPD with p ε U and q ε V.
Need to show: pq is the closest pair for (U,V).

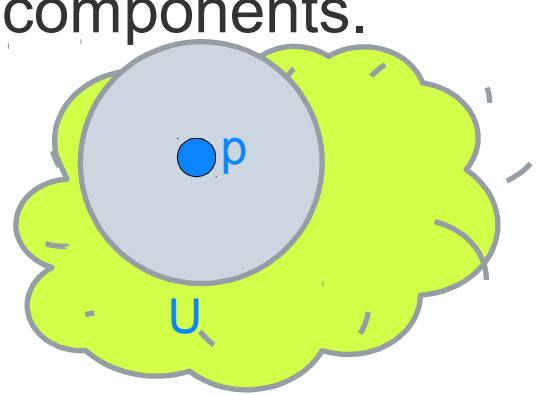


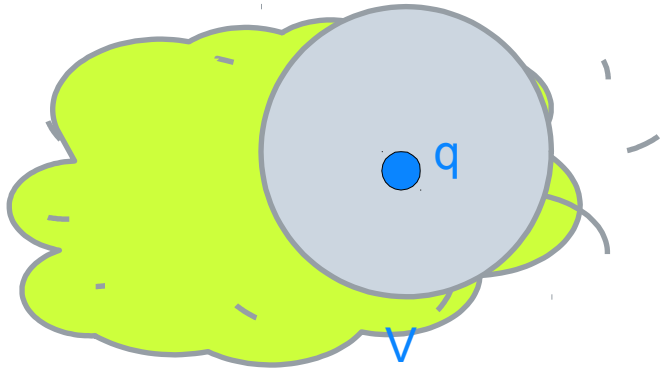


WSPD and EMST

Lemma: Let {(U1,V1), (U2, V2), …, (Um, Vm)} be an ε-WSPD for P, and let G be the graph on P that for each i has an edge between the closest pair between Ui and Vi. Then G contains the EMST of P.

Proof: Need to show: pq is the closest pair for (U,V).
Consider an execution of Kruskal's algorithm on P.
When pq is considered, p and q are in different components.

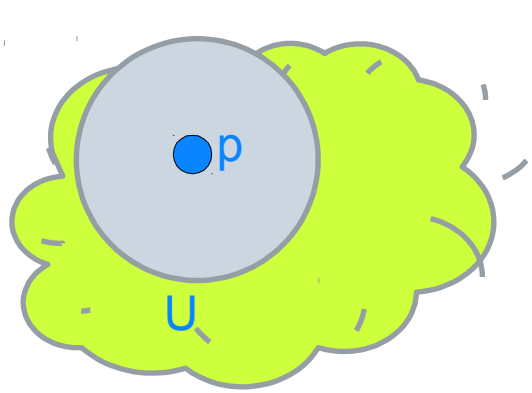


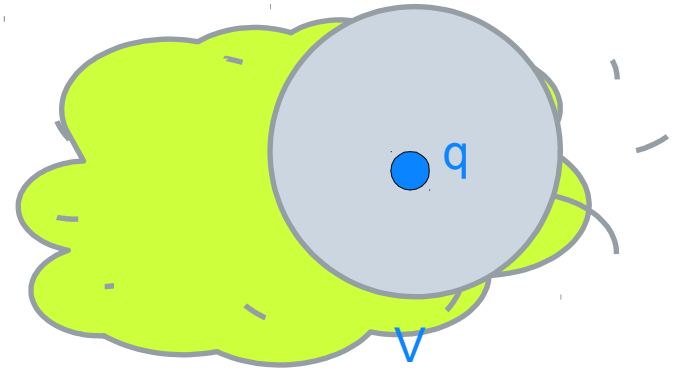


WSPD and EMST

Lemma: Let {(U1,V1), (U2, V2), …, (Um, Vm)} be an ε-WSPD for P, and let G be the graph on P that for each i has an edge between the closest pair between Ui and Vi. Then G contains the EMST of P.

Proof: These components wholly contain U and V, respectively, because of well-separation.

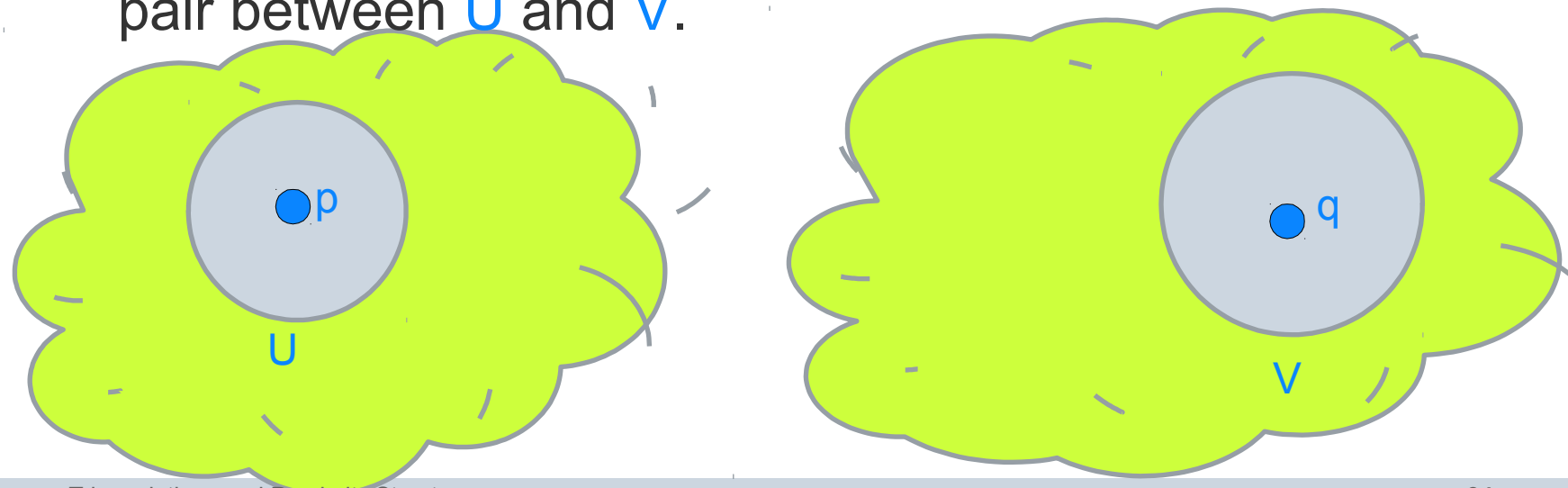




WSPD and EMST

Lemma: Let {(U1,V1), (U2, V2), …, (Um, Vm)} be an ε-WSPD for P, and let G be the graph on P that for each i has an edge between the closest pair between Ui and Vi. Then G contains the EMST of P.

Proof: Hence, pq is the first edge between U and V considered by Kruskal's algorithm, so it is the closest pair between U and V.

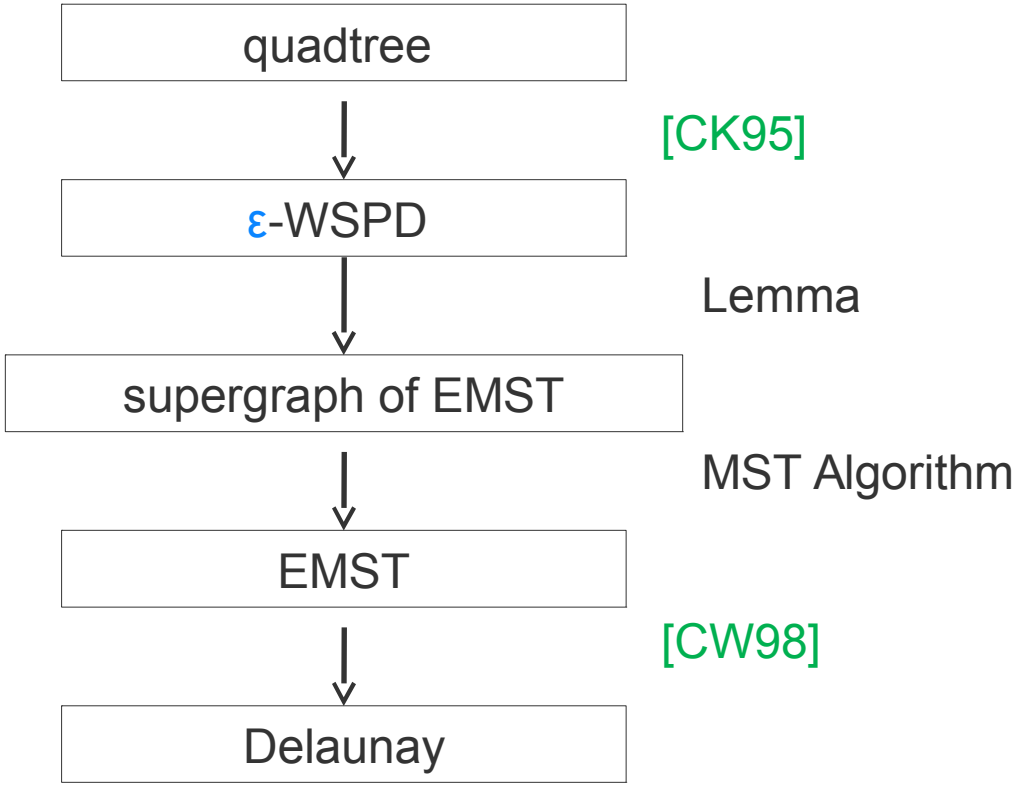


Algorithm Outline

The lemma leads to the following strategy:

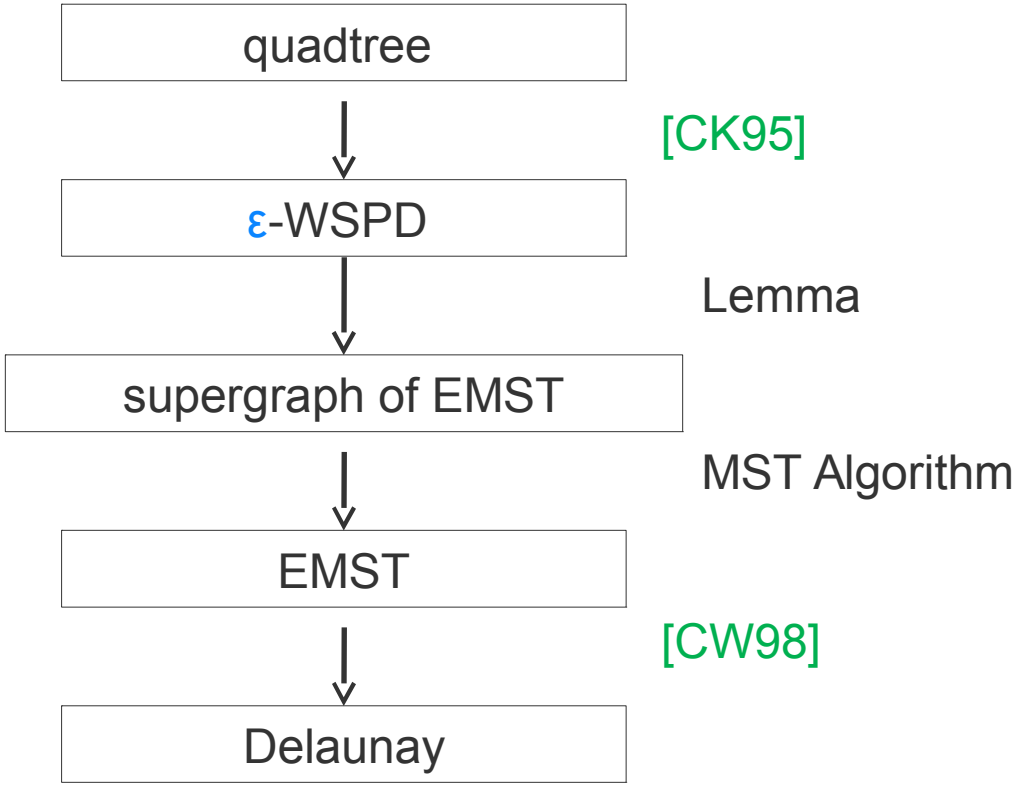
Given: compressed quadtree for P.

Want: Delaunay triangulation for P.



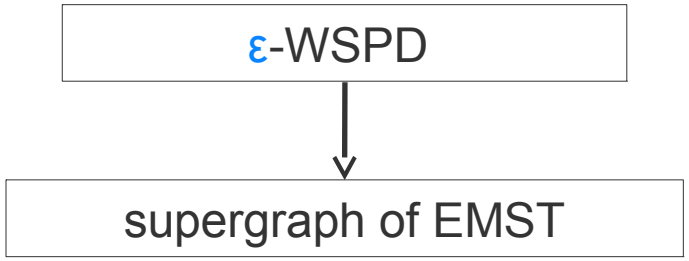
Implementing the Strategy

We need to overcome several challenges.



Implementing the Strategy

To make this work, we need to overcome several challenges.



By the lemma, we can find the a sparse sugergraph of the EMST by solving a sequence of bichromatic closest pair problems.

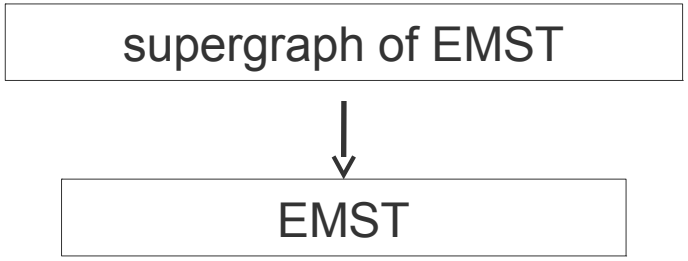
But how to do that?

The total size of the sets Ui, Vi may be quadratic.

Finding the closest pair between Ui and Vi usually takes O((|Ui|+|Vi|)log (min (|Ui|, |Vi|))) time.

Implementing the Strategy

To make this work, we need to overcome several challenges.

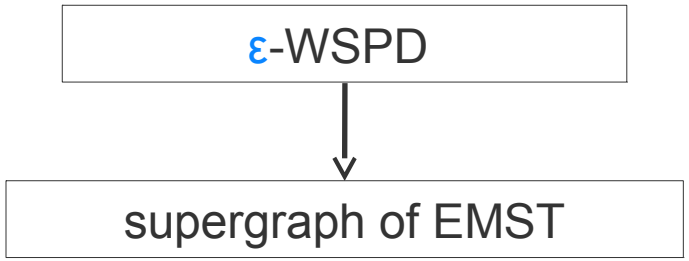


We need to find the MST of the sparse supergraph.

The fastest known deterministic MST-algorithm for the pointer machine whose running time can be analyzed runs in time $O(n\alpha(n))$. [C00]

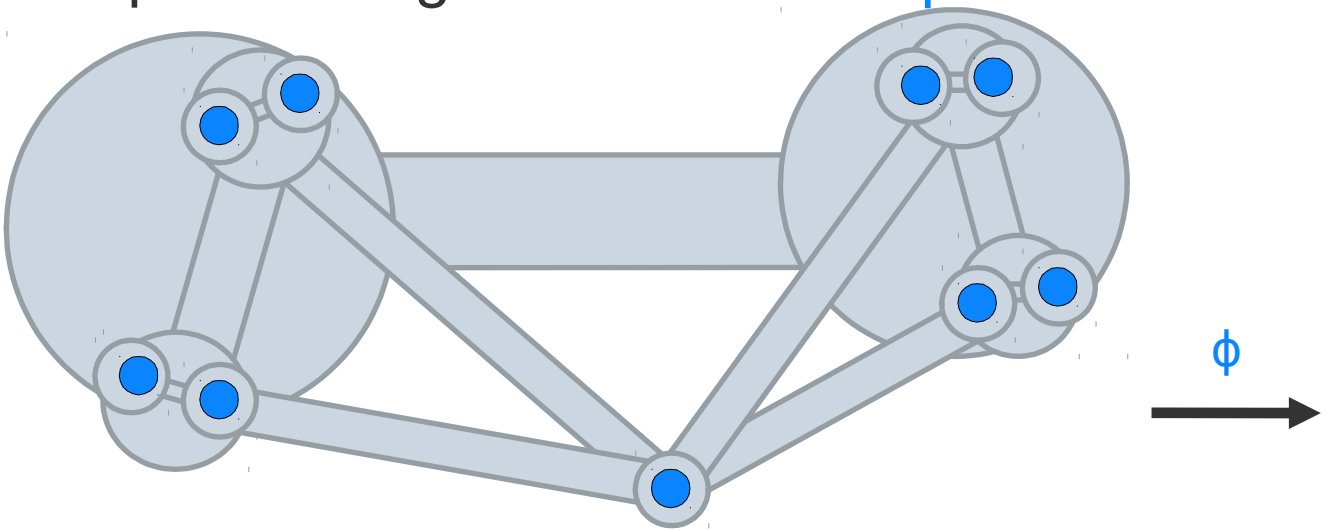
For planar graphs, Borůvka's algorithm needs only linear time, but the supergraph need not be planar.

Reducing the Weight

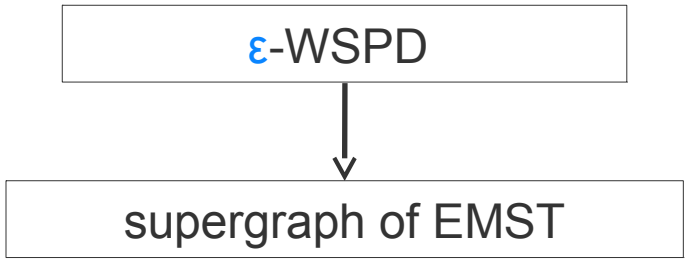


First, we need to reduce the total size of the Ui, Vi sets.

Similar to Yao [1982], we partition the pairs by their "general direction" and process each part individually.
Take all WSPD-pairs with general direction ϕ.

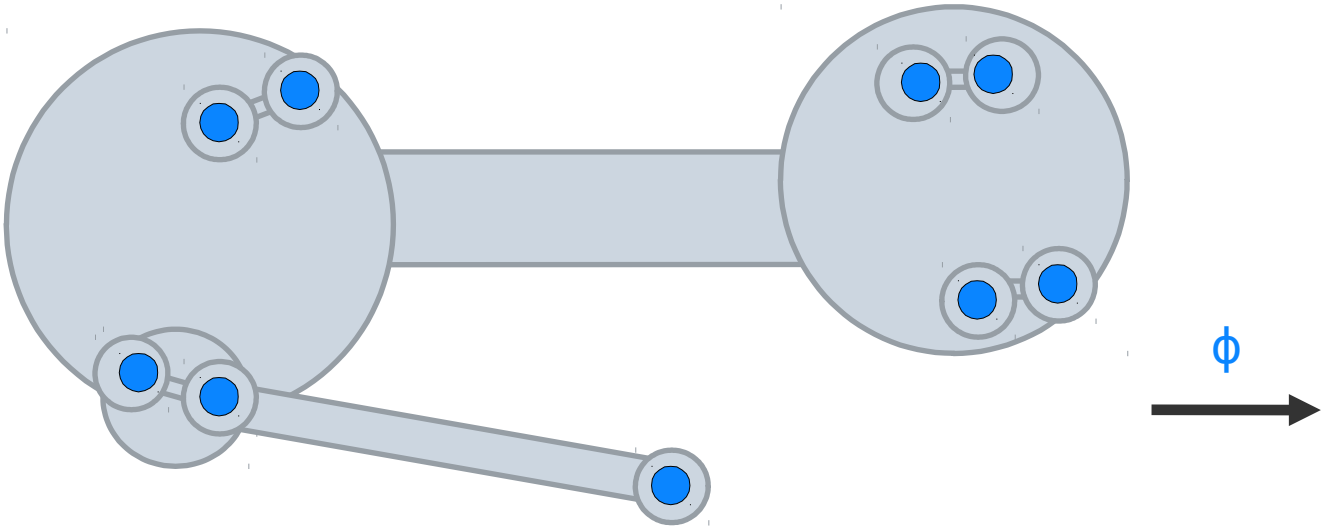


Reducing the Weight

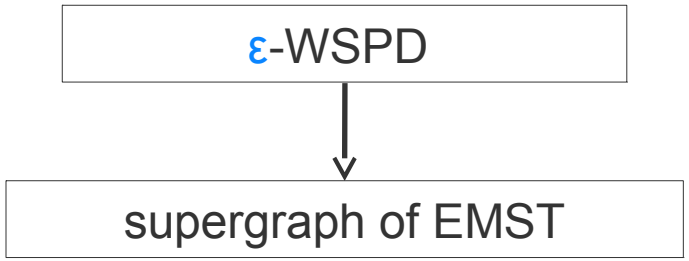


First, we need to reduce the total size of the Ui, Vi sets.

For every point find the k closest pairs in each direction ϕ and remove it from all other pairs.

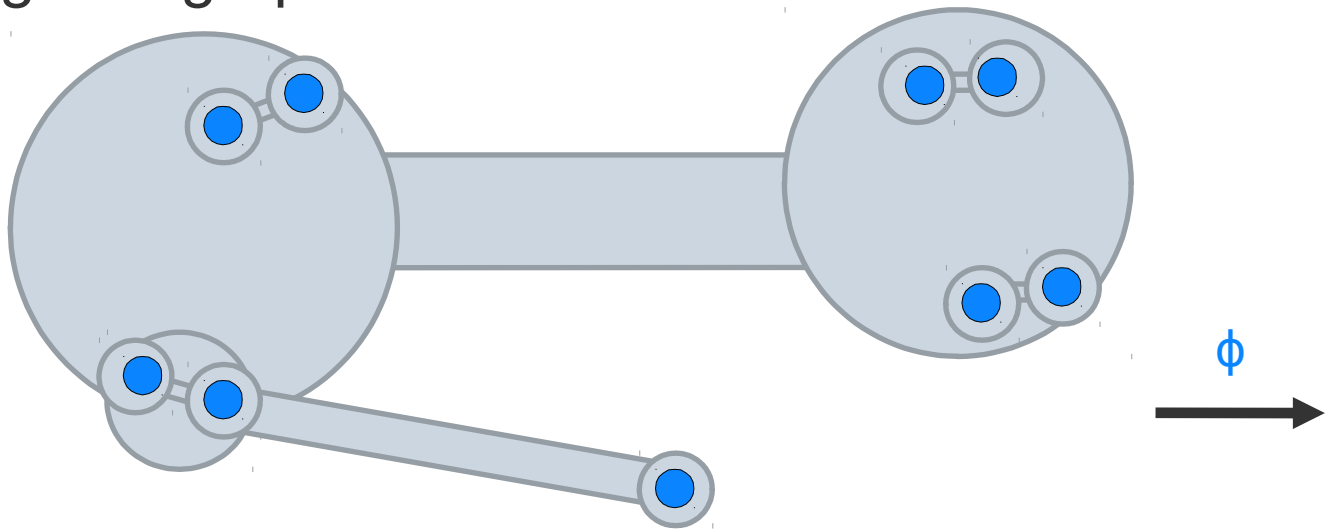


Reducing the Weight

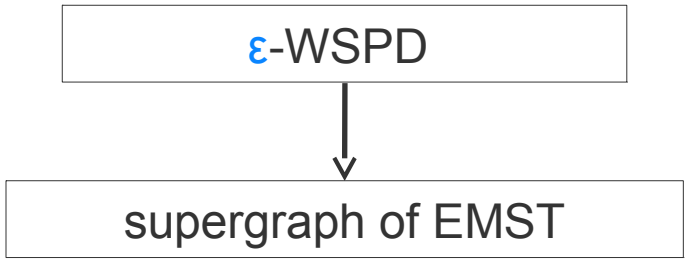


First, we need to reduce the total size of the Ui, Vi sets.

Lemma: This can be done in linear time and results in a collection of pairs with total linear size whose bichromatic nearest neighbor graph still contains the EMST.

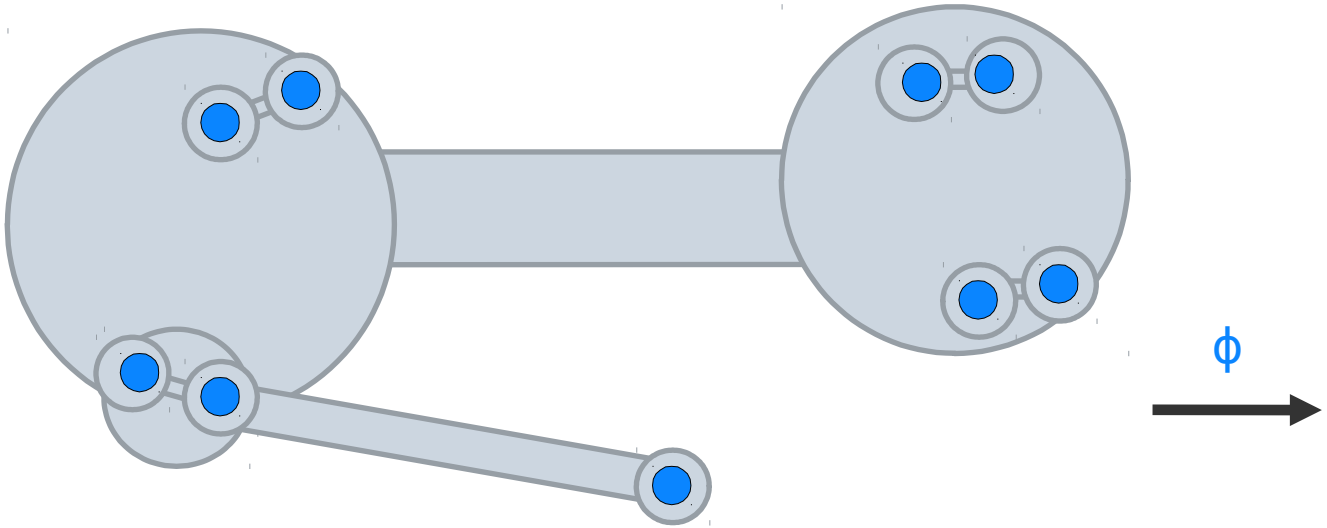


Finding the Nearest Neighbors

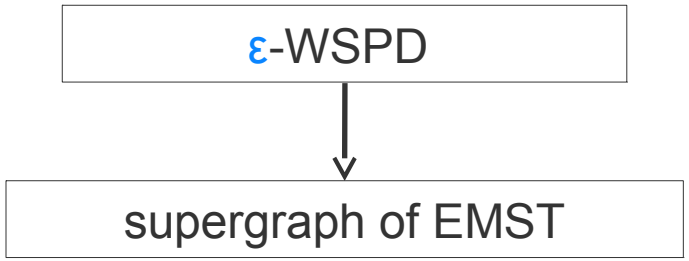


Second, we need to find the bichromatic nearest neighbors for each reduced pair.

If the points were sorted in in the right order (perpendicular to ϕ), this could be done in linear time.

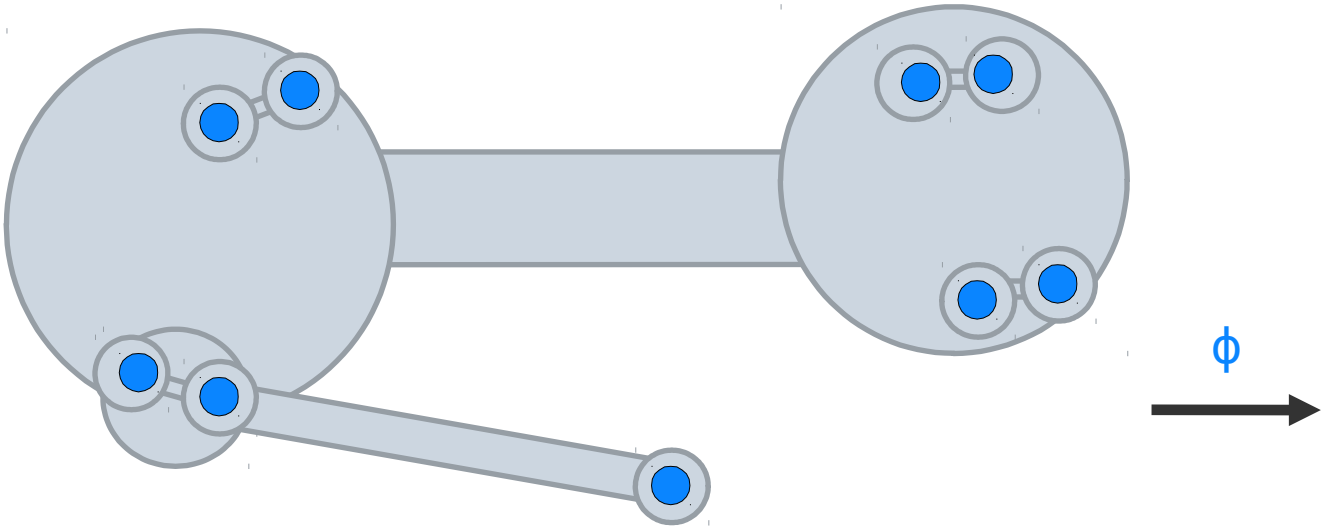


Finding the Nearest Neighbors

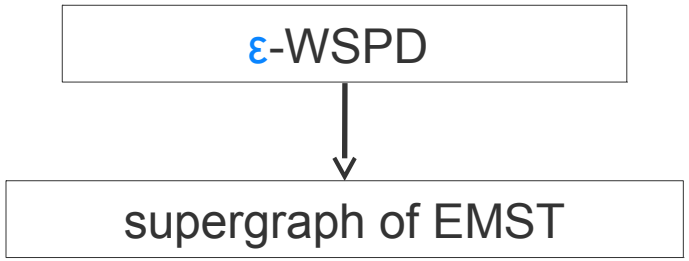


Second, we need to find the bichromatic nearest neighbors for each reduced pair.

However, sorting would take too much time.

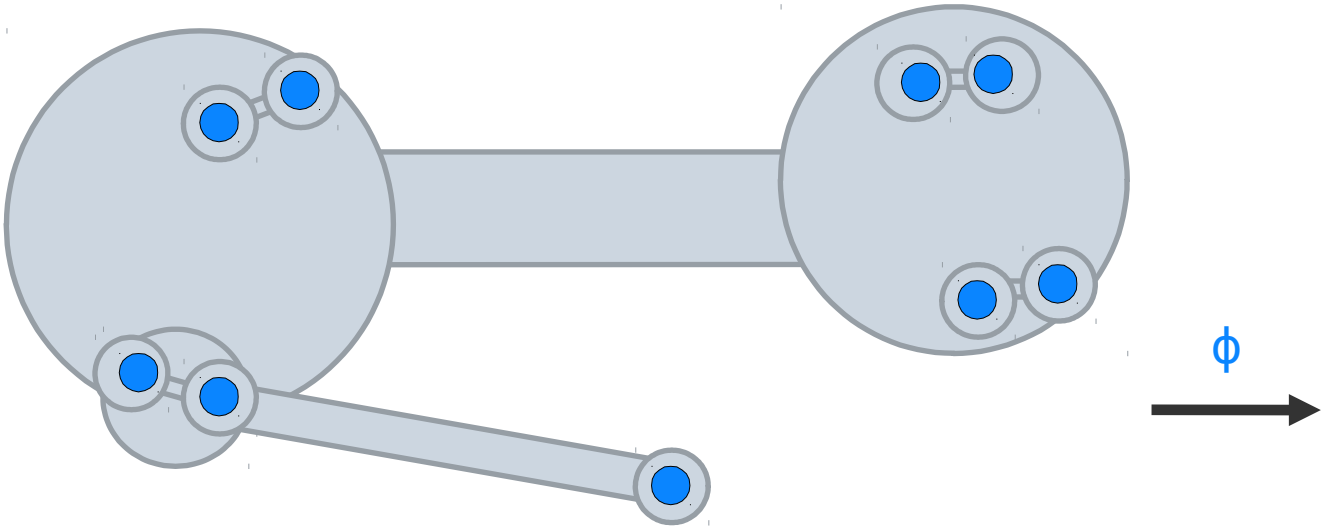


Finding the Nearest Neighbors

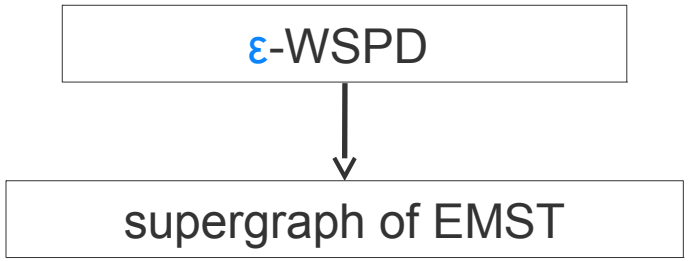


Second, we need to find the bichromatic nearest neighbors for each reduced pair.

But we only need to sort locally, so that the order for the points in each pair is known.

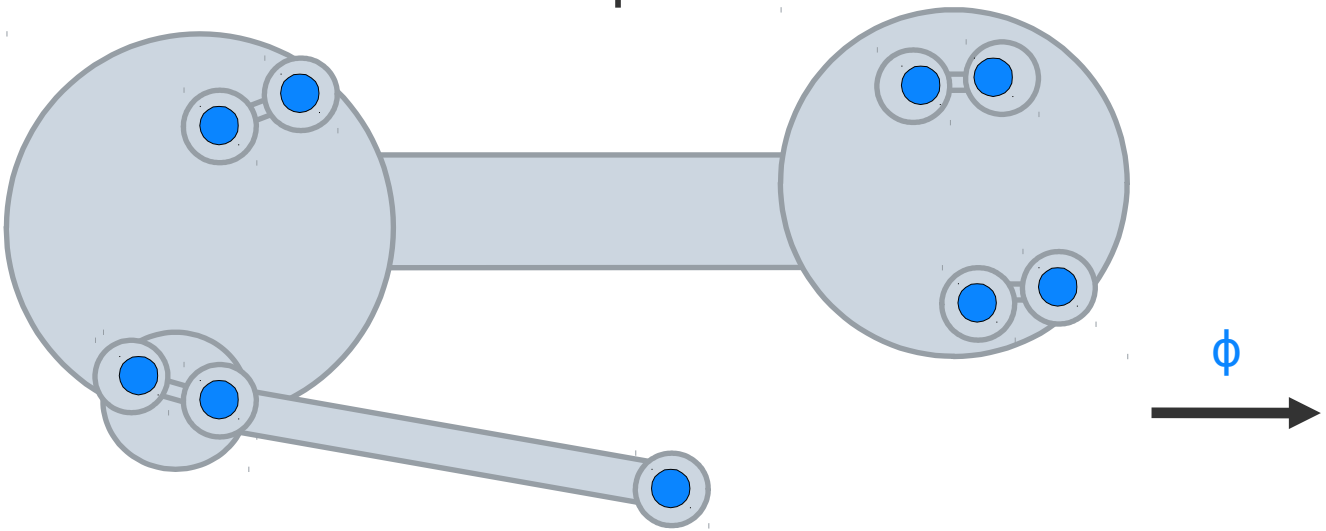


Finding the Nearest Neighbors

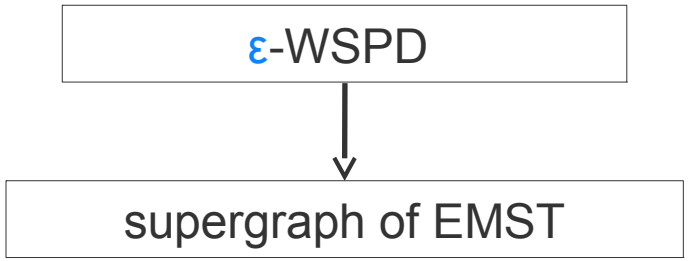


Second, we need to find the bichromatic nearest neighbors for each reduced pair.

We can use the structure of the quadtree to define a directed graph H of linear size, such that a topological ordering of H yields the desired order for each pair.

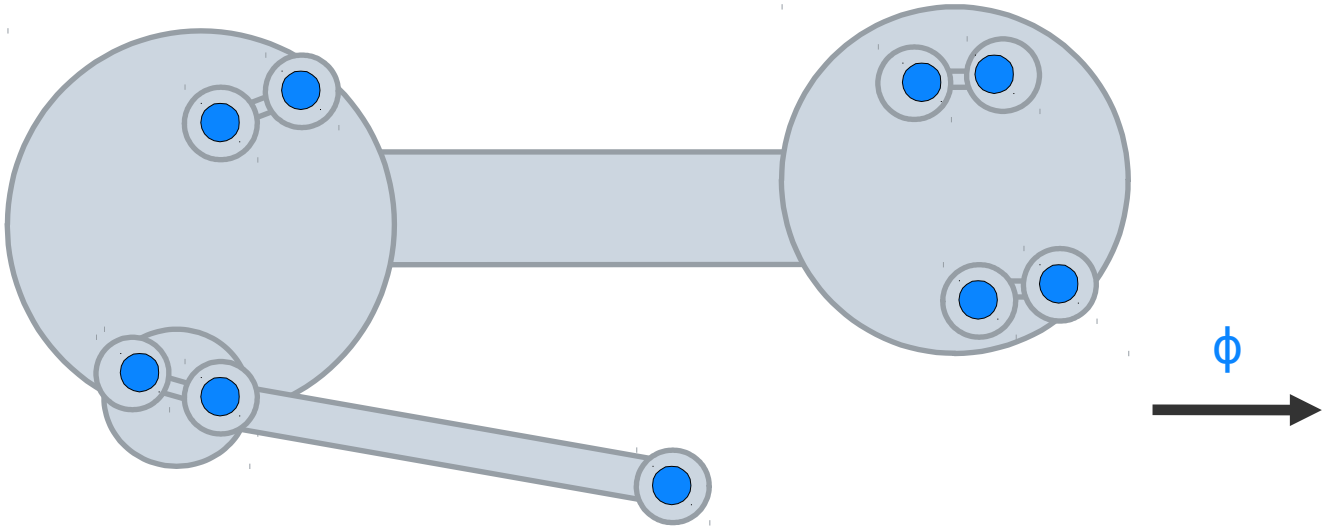


Finding the Nearest Neighbors

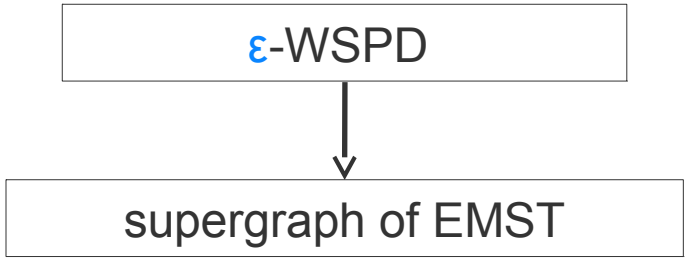


Second, we need to find the bichromatic nearest neighbors for each reduced pair.

A topological ordering can be found in O(n) time via depth first search.



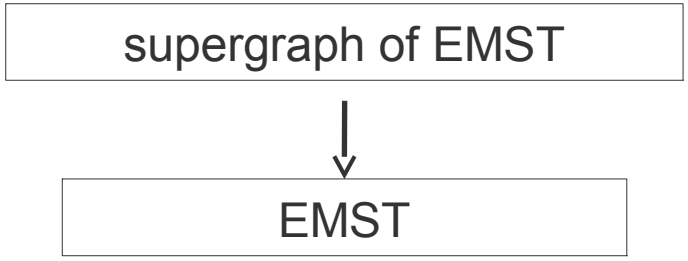
Finding the Nearest Neighbors



To summarize, the supergraph of the EMST is found in three steps.

1. Prune the pairs of the WSPD to make their size linear.
2. Locally sort the points in each pair through a topological sort of an appropriate graph.
3. Use the local ordering to find the nearest neighbors.

Finding the EMST



Finally, we need to extract the MST.

This can be done in linear time by an appropriate variant of Borůvka's algorithm that exploits the structure of the compressed quadtree.

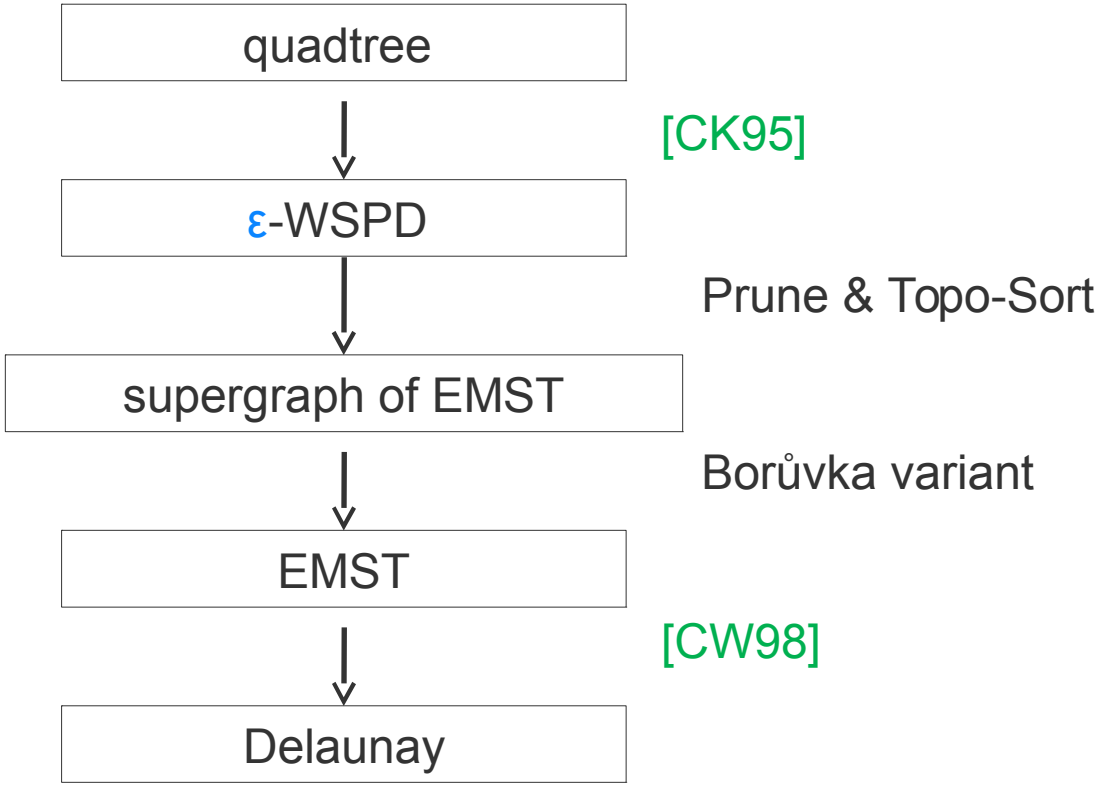
We use the crossing number inequality to bound the number of edges we need to consider.

Algorithm Outline

This finally concludes the description of the algorithm.

Given: a compressed quadtree for P.

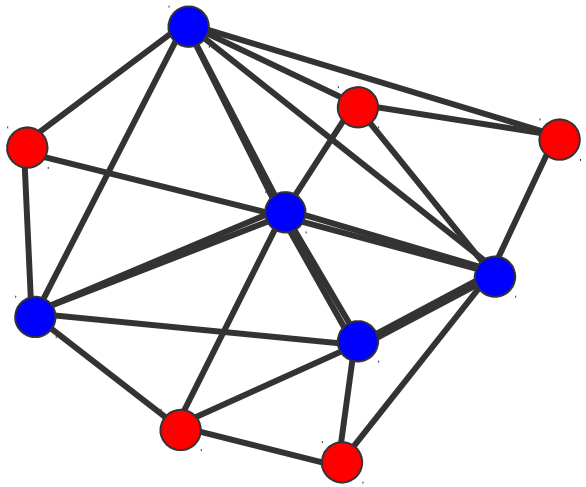
Want: the Delaunay triangulation for P.



Applications

Applications

Splitting Delaunay triangulations
Given the DT of a bicolored n-point set, we can compute the DT of the blue points in O(n) expected time [CDHMST02] [CM09].



Applications

Splitting Delaunay triangulations
Given the DT of a bicolored n-point set, we can compute the DT of the blue points in O(n) expected time [CDHMST02] [CM09].

Now we can do it deterministically.

Convert the DT into a quadtree, remove the red points from the tree (easy), convert the pruned quadtree back into a DT.

Applications

Transdichotomous Delaunay triangulations
Using randomized approach and bit tricks the Delaunay triangulation of a planar point set can be found in O(sort(n)) expected time on a word RAM.

Applications

Transdichotomous Delaunay triangulations
Using randomized approach and bit tricks the Delaunay triangulation of a planar point set can be found in O(sort(n)) expected time on a word RAM.

Now we can do it deterministically in O(n loglog n) time.

Applications

and a few more…

- preprocessing planar regions for Delaunay triangulations
- self-improving algorithms for planar DTs
- …

Open Problems

Can our algorithm be simplified?

Can we find a deterministic algorithm for splitting 3D convex hulls?

Are there further relationships between proximity structures to be discovered?

Questions?