

Meshes and Nets

Don Sheehy

Joint work with Gary Miller and Todd Phillips

A **new** meshing algorithm.

$O(n \log n + m)$ - Optimal
- Output-sensitive

A **new** meshing algorithm.

$O(n \log n + m)$ - Optimal
- Output-sensitive

A more general view of the meshing problem.

A **new** meshing algorithm.

$O(n \log n + m)$ - Optimal
- Output-sensitive

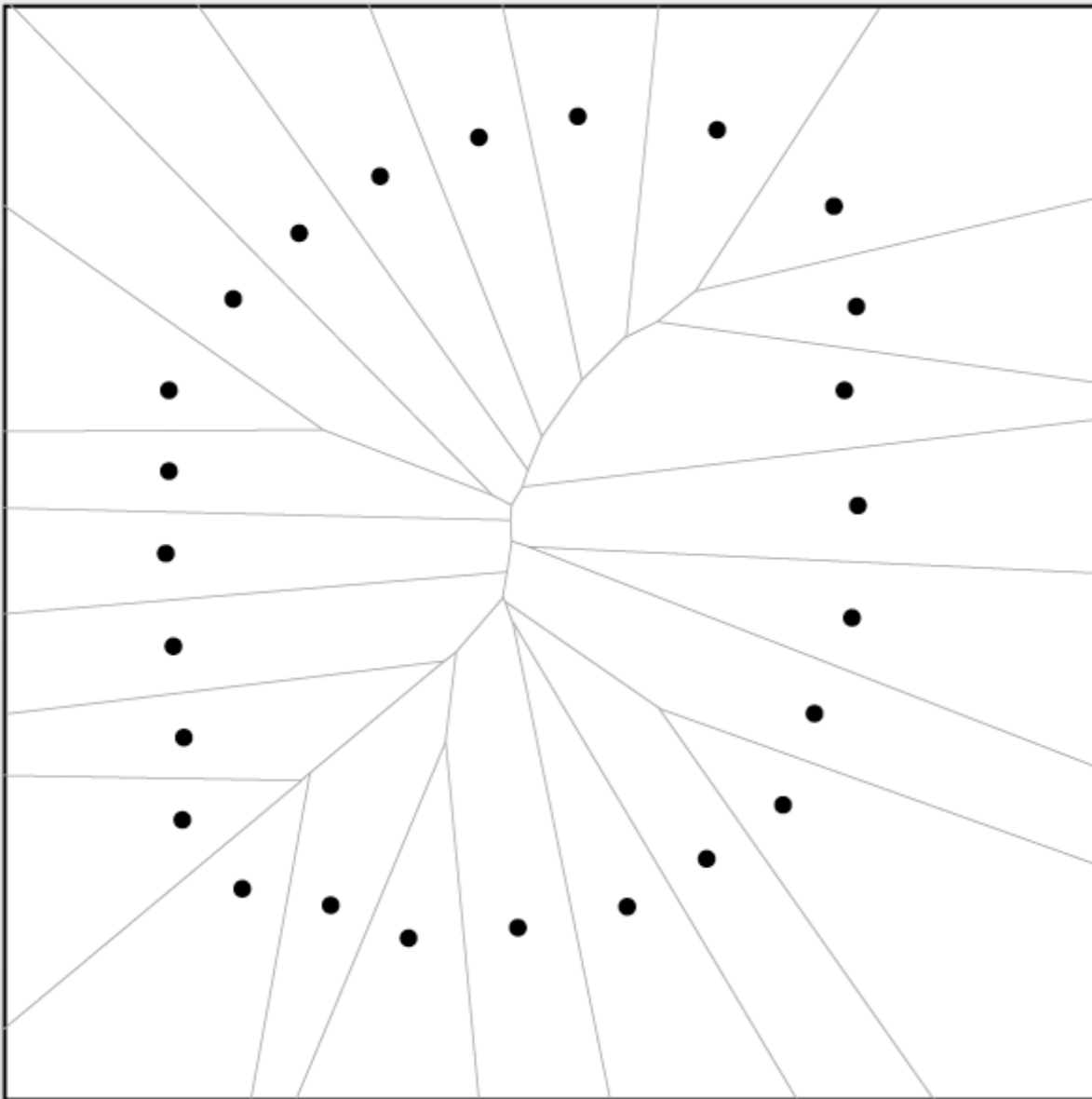
A more general view of the meshing problem.

Some nice theory.

Meshing

Input: $P \subset \mathbb{R}^d$

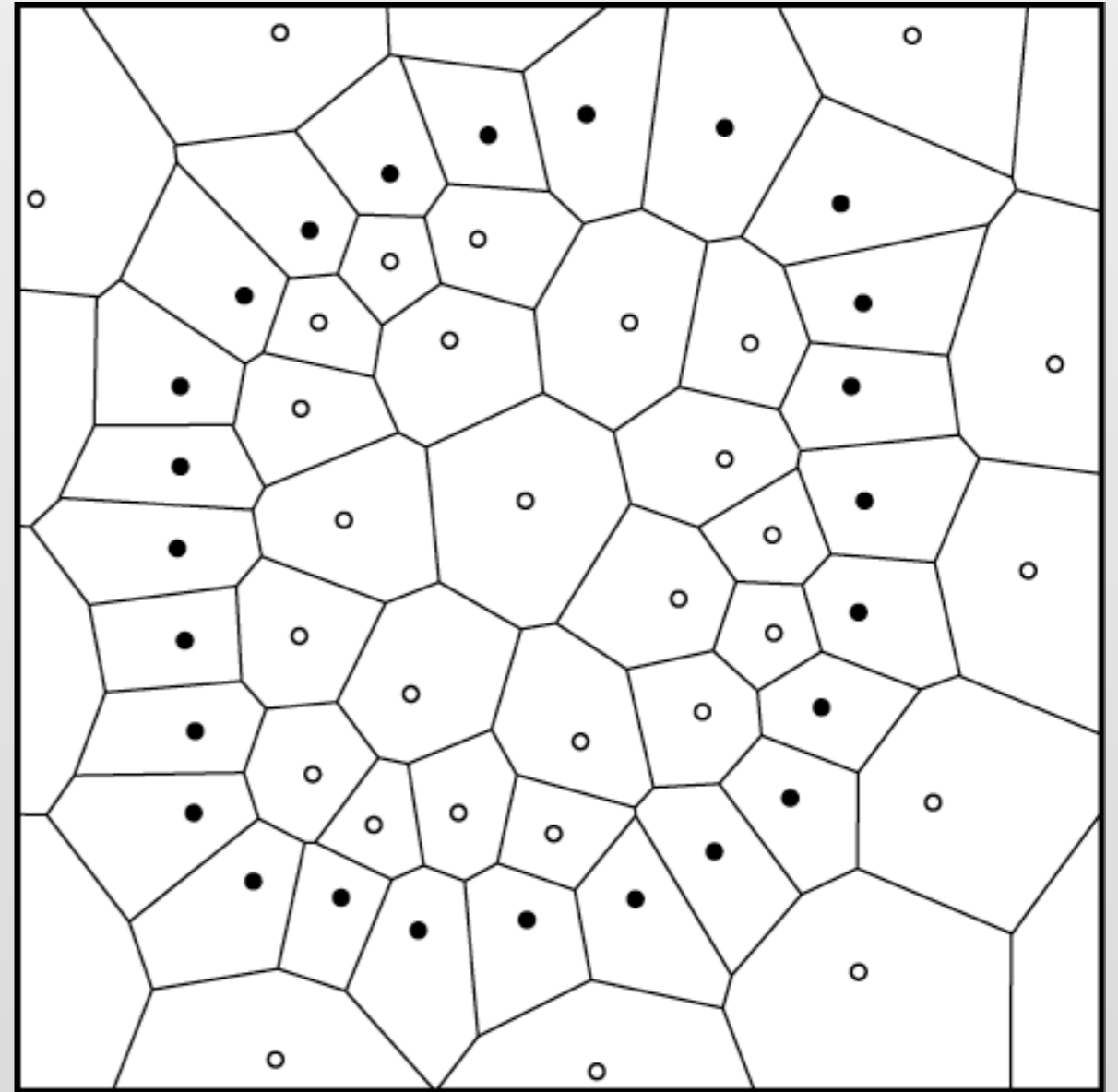
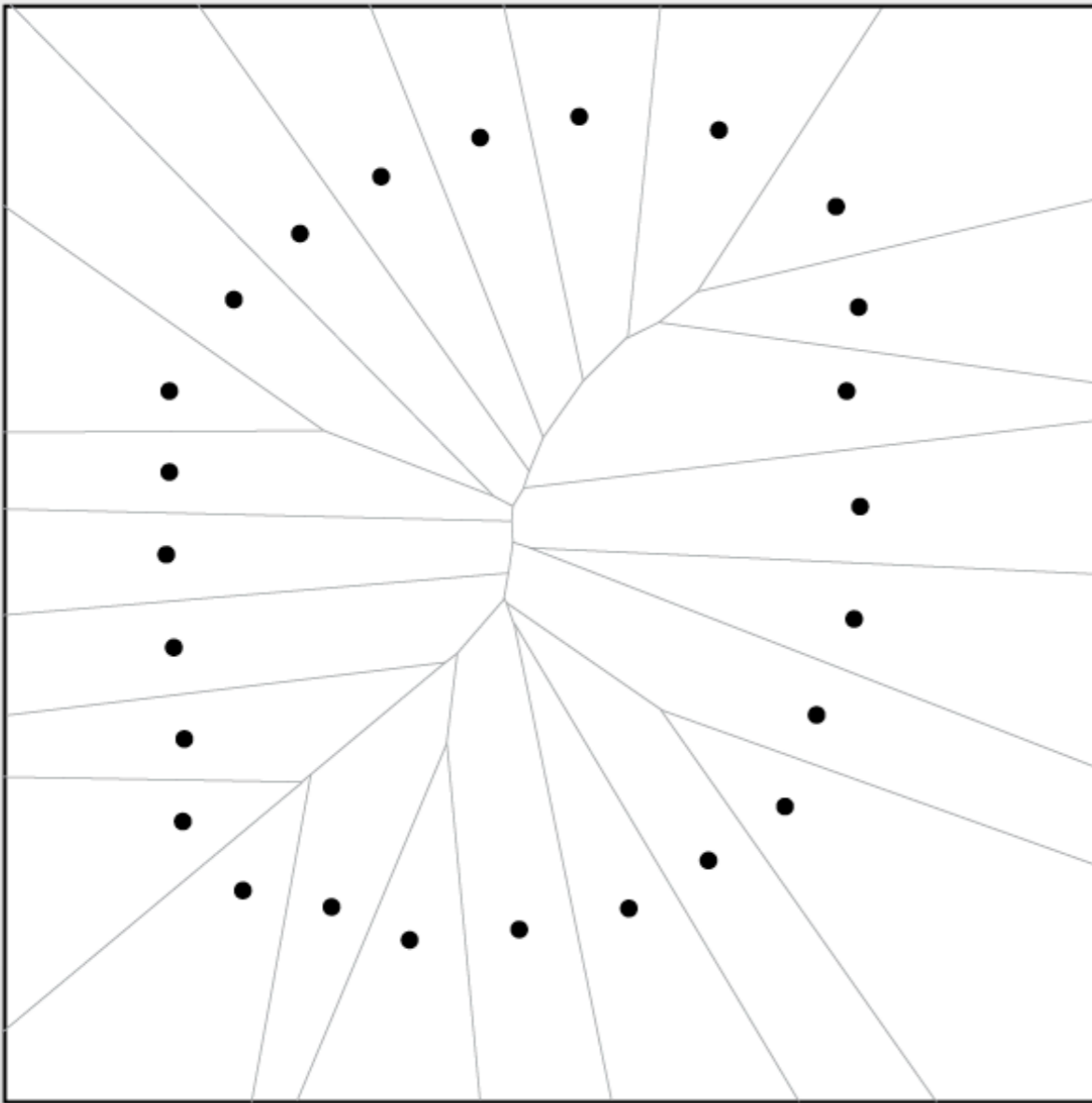
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

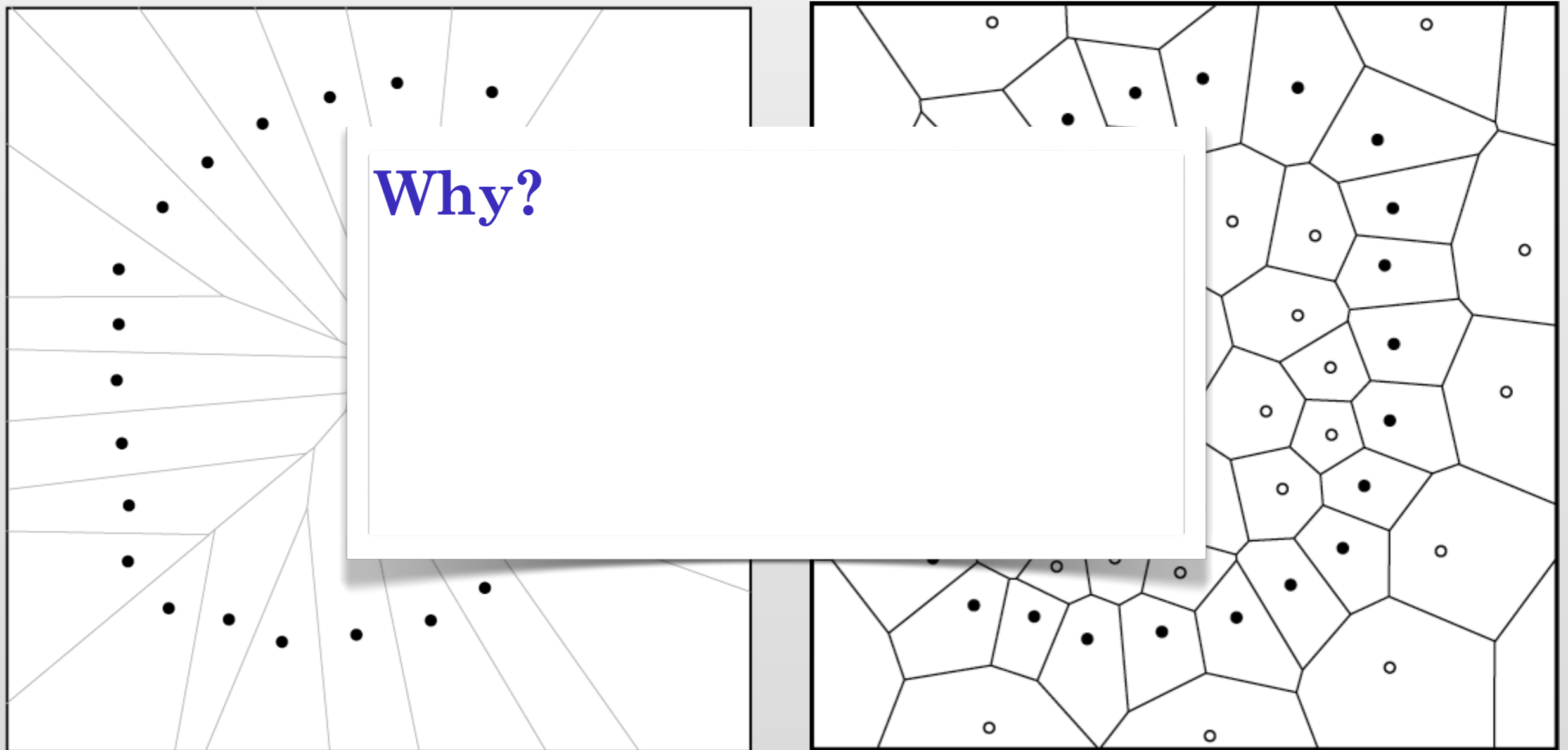
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

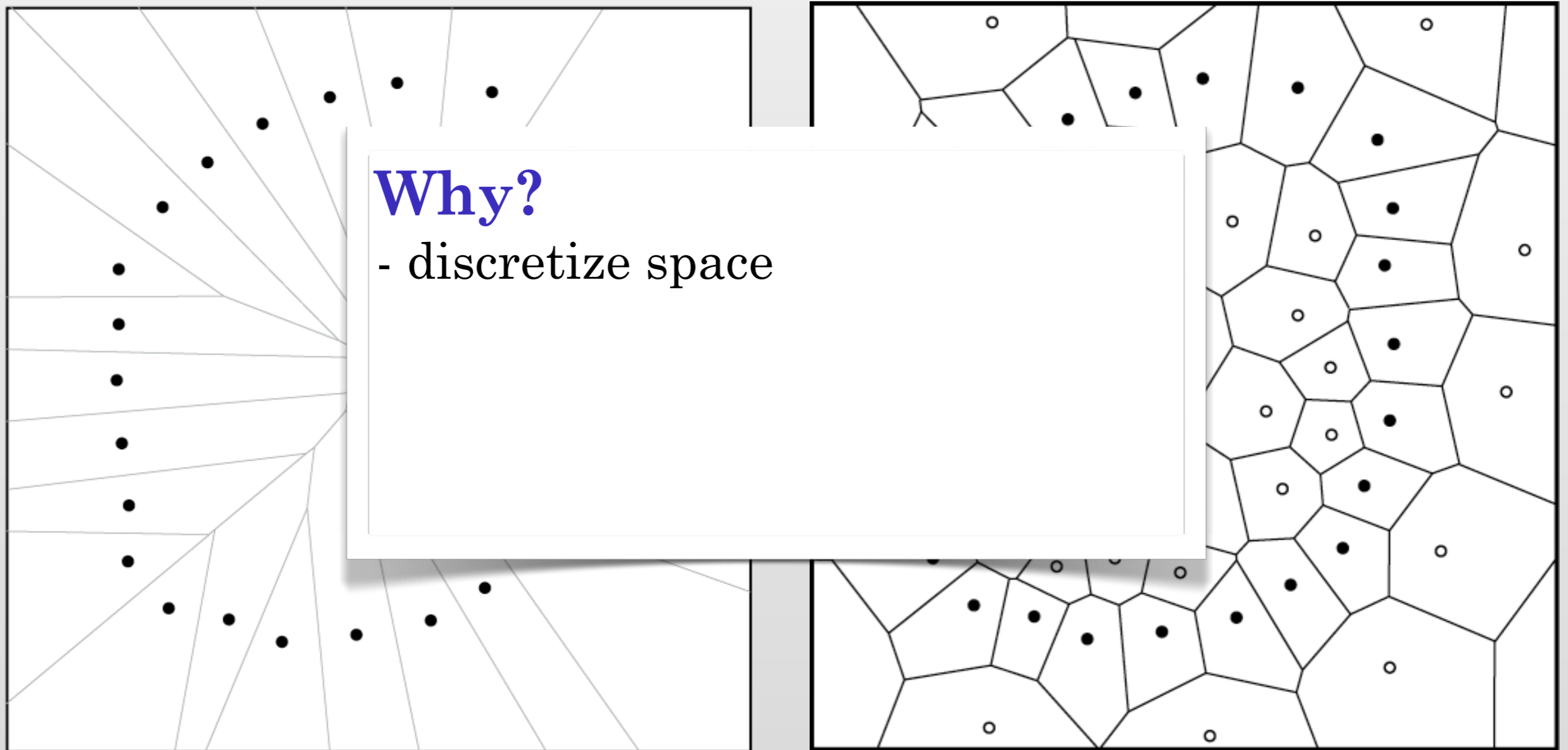
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

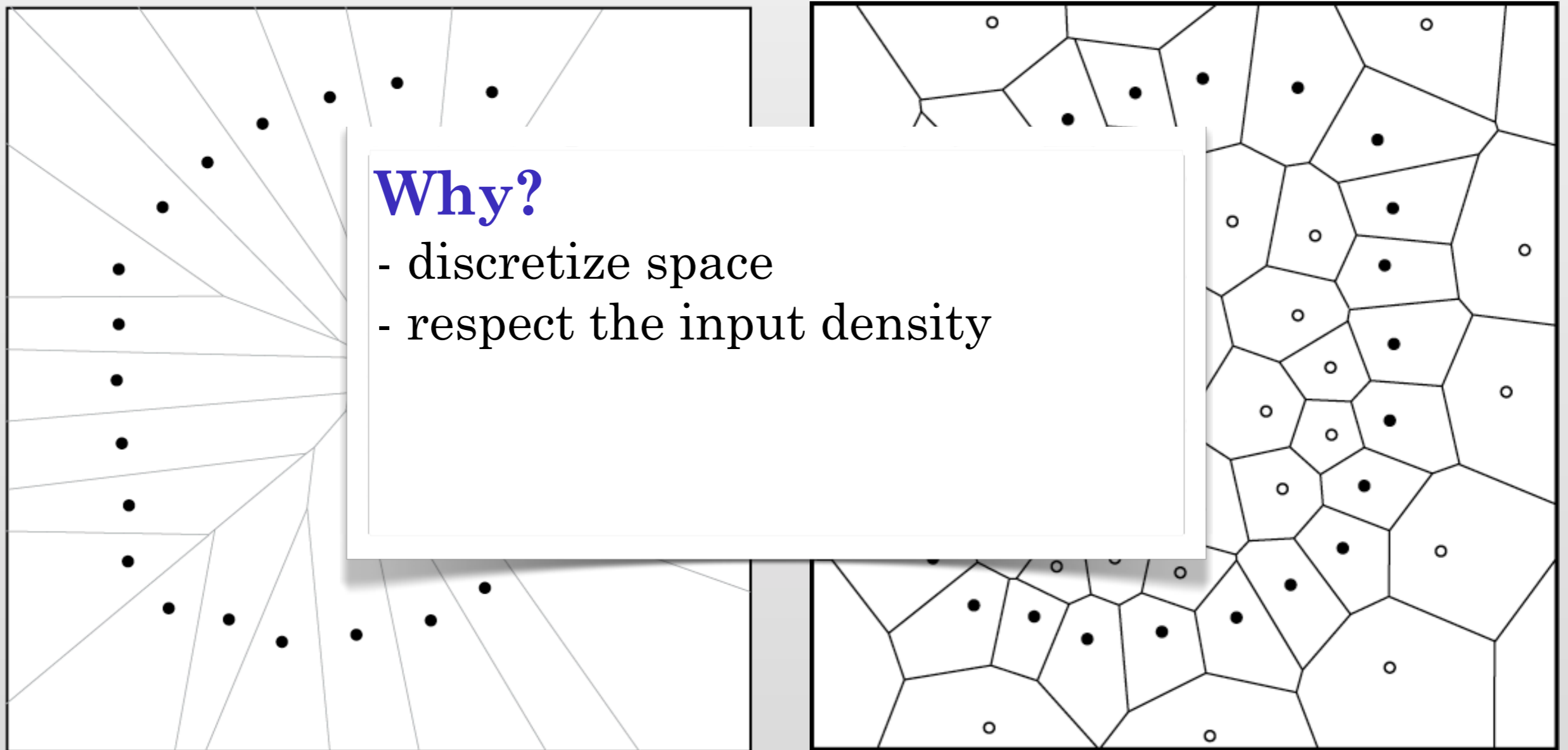
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

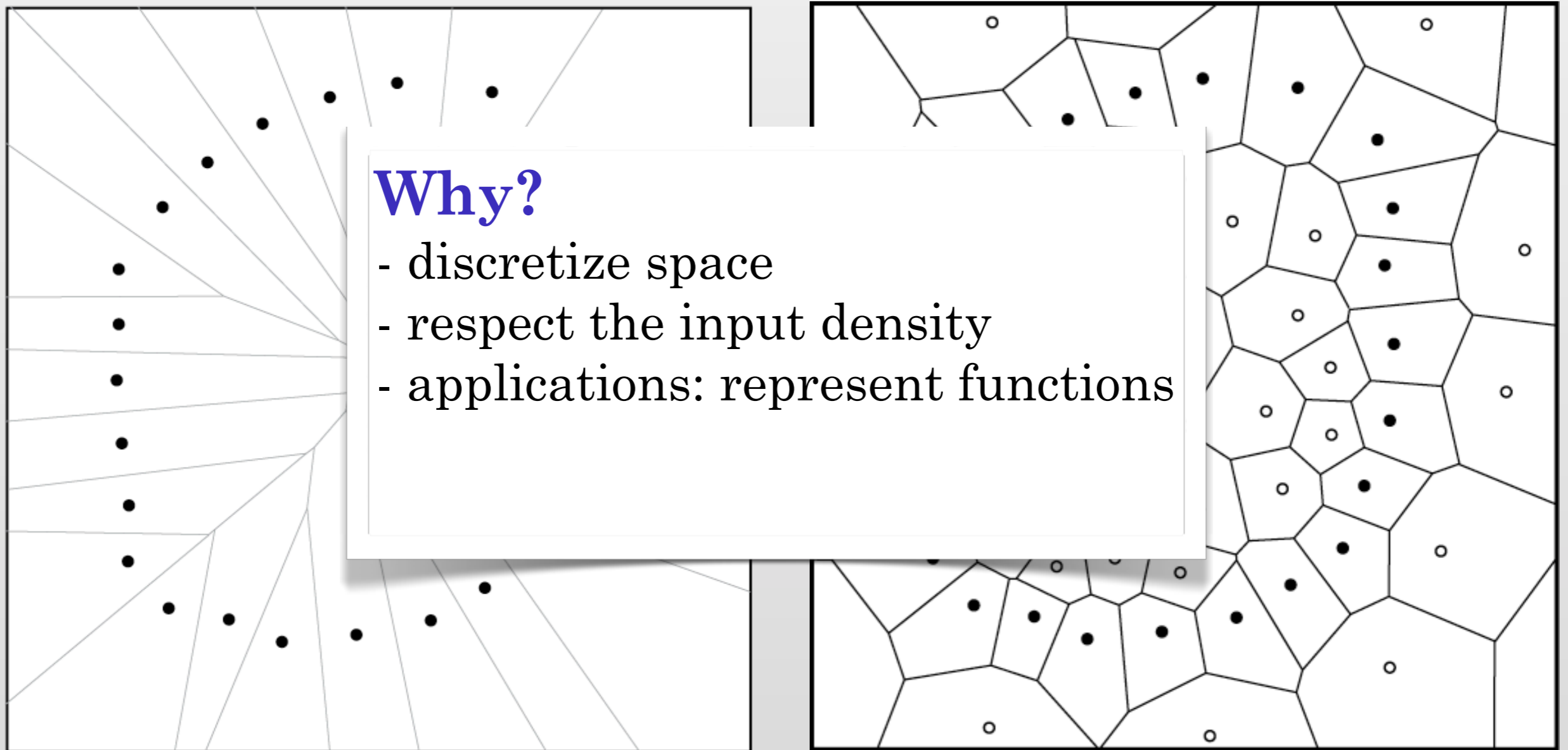
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

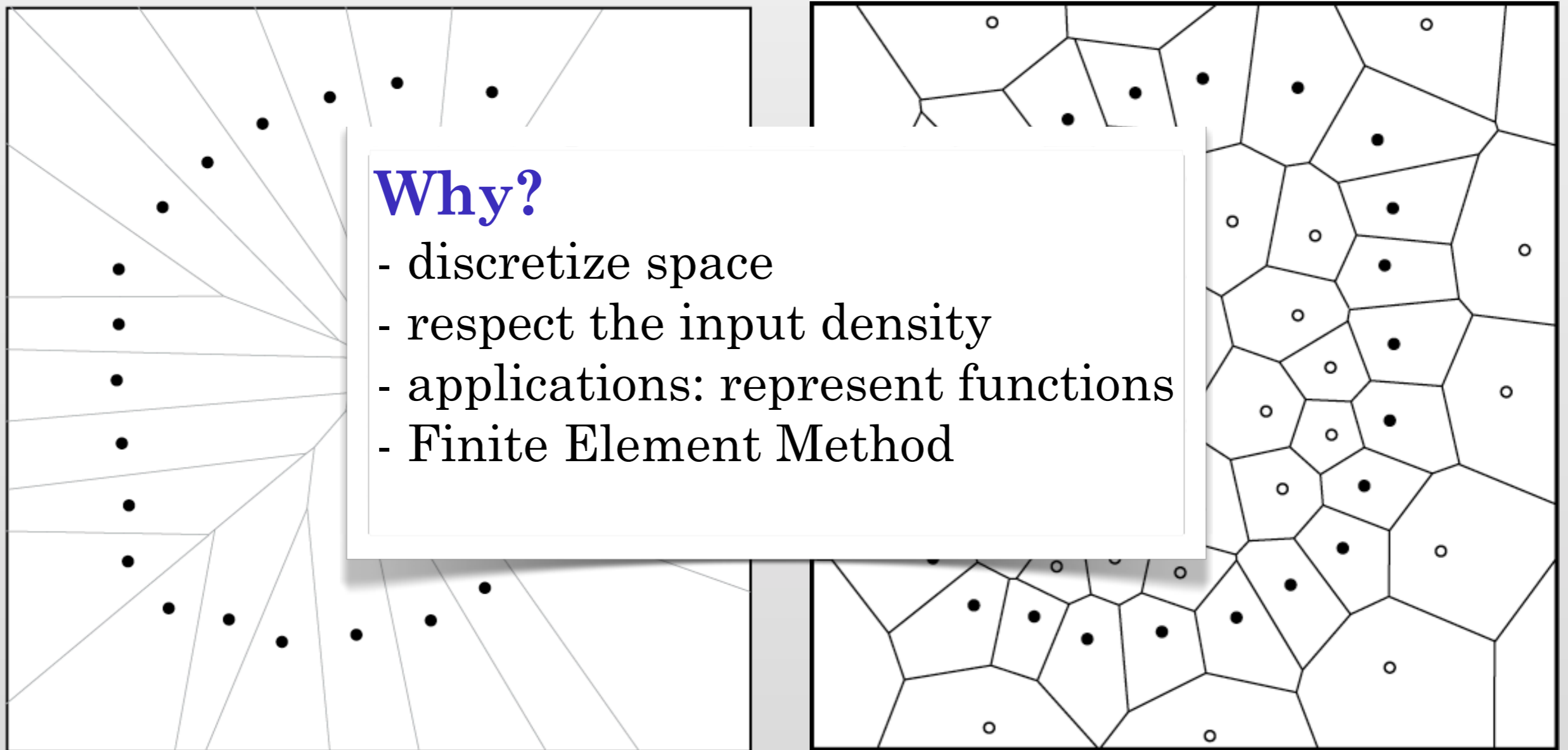
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

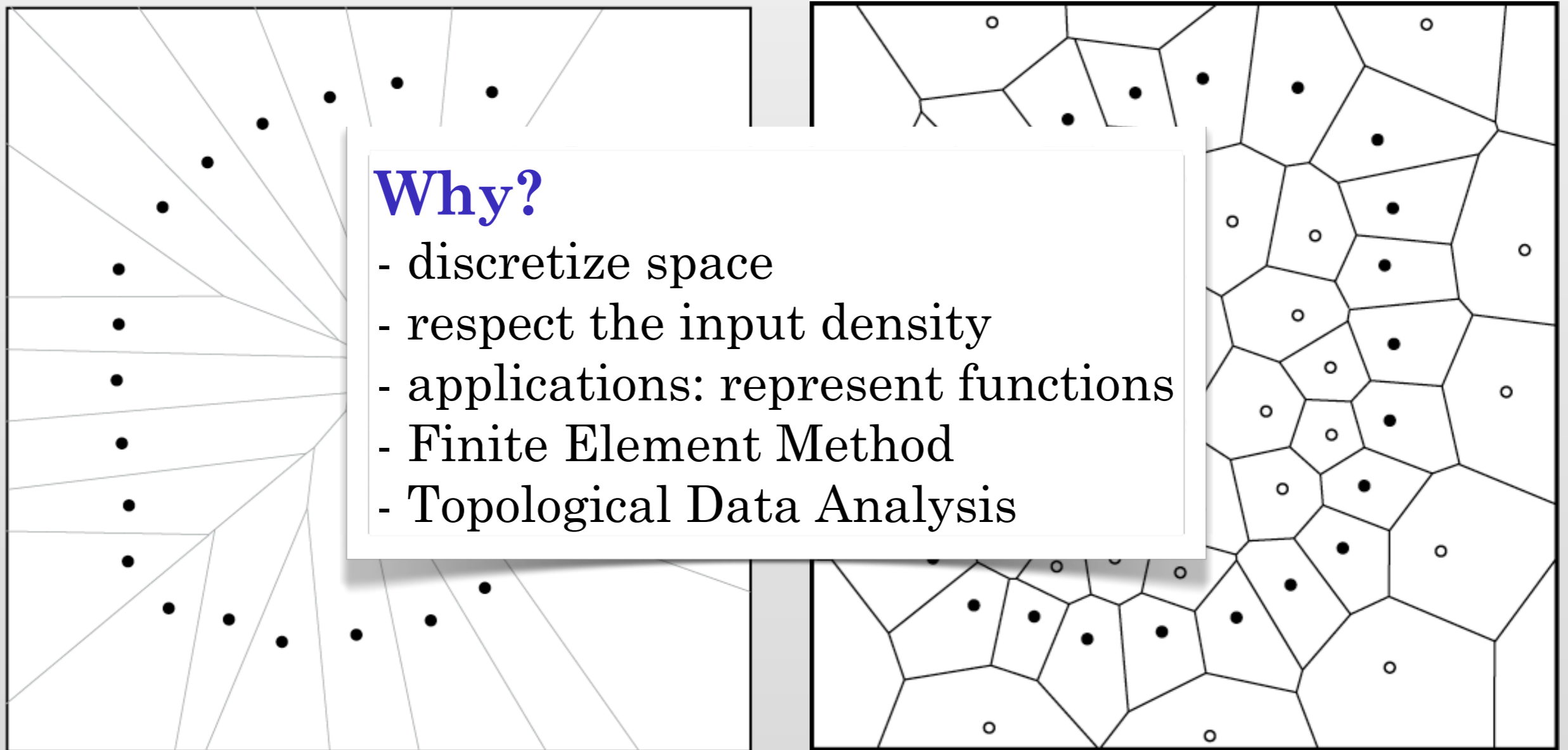
Output: $M \supset P$ with a “nice” Voronoi diagram



Meshing

Input: $P \subset \mathbb{R}^d$

Output: $M \supset P$ with a “nice” Voronoi diagram



Overview

Algorithm

Algorithm

1 Order the inputs

Algorithm

- 1 Order the inputs
- 2 Incremental Updates

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)
- 2 Hierarchical Quality

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)
- 2 Hierarchical Quality
- 3 Quality \Rightarrow low complexity

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)
- 2 Hierarchical Quality
- 3 Quality \Rightarrow low complexity
- 4 Range Space Nets

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)
- 2 Hierarchical Quality
- 3 Quality \Rightarrow low complexity
- 4 Range Space Nets
- 5 Ball Cover Lemma

Algorithm

- 1 Order the inputs
- 2 Incremental Updates
- 3 Refinement
- 4 Point Location
- 5 Finishing

Analysis

- 1 Classic Mesh Size Analysis
(with a twist)
- 2 Hierarchical Quality
- 3 Quality \Rightarrow low complexity
- 4 Range Space Nets
- 5 Ball Cover Lemma
- 6 Amortize Point Location Cost

Metric Nets

“Nets catch everything that’s big.”

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

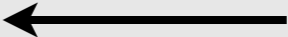
1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$  **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$  **Packing**

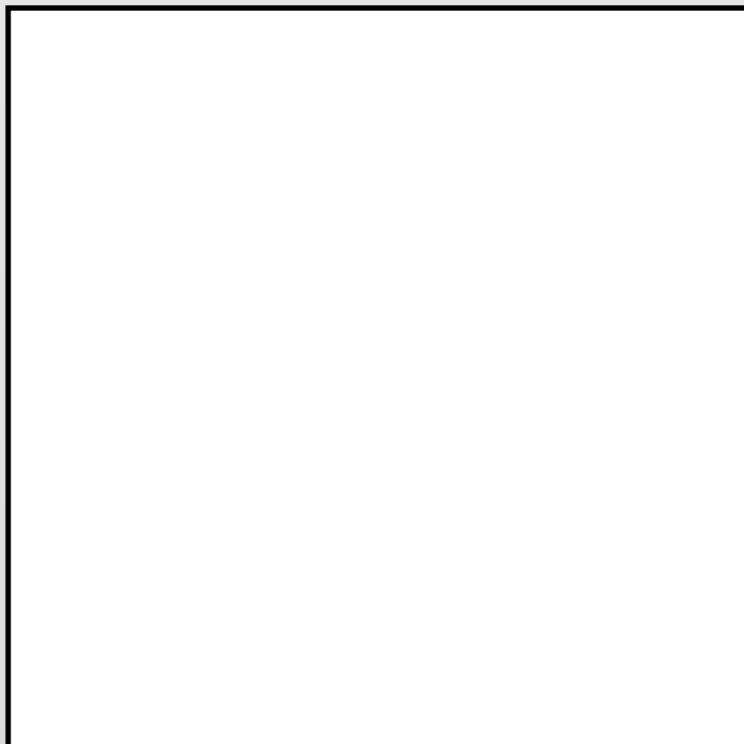
2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$  **Covering**

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$ ← **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$ ← **Covering**

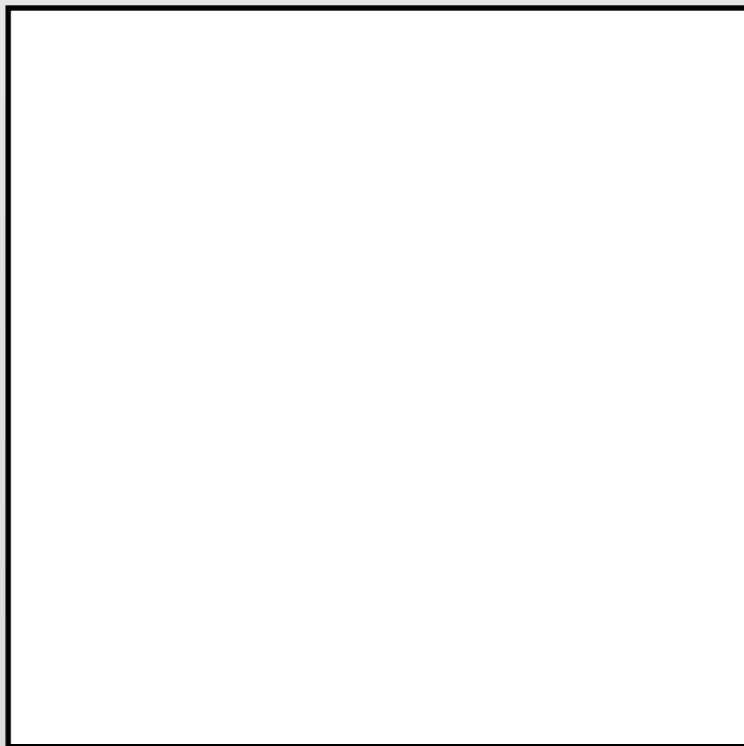


“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$ ← **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$ ← **Covering**



Example: Bounded Euclidean Space

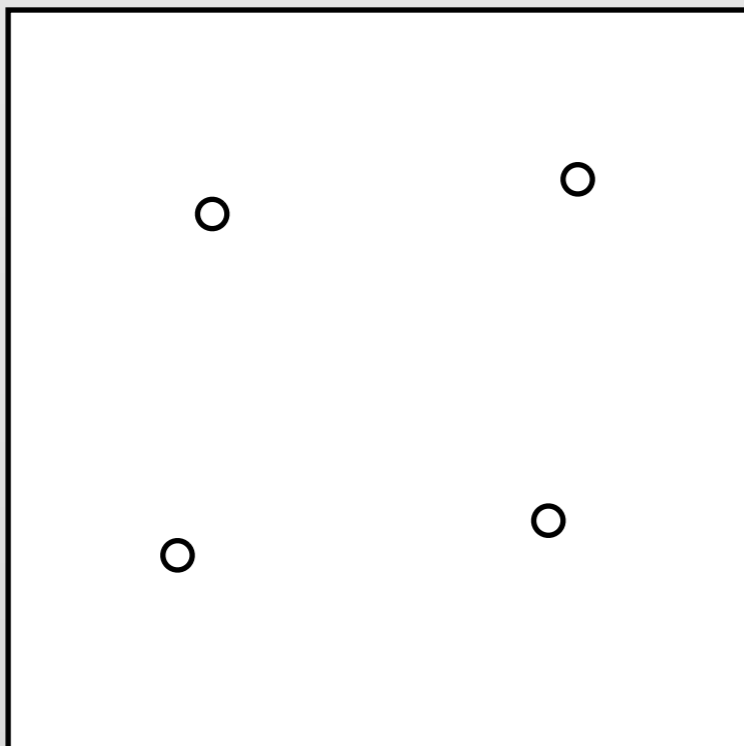
Metric Nets

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$ ← **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$ ← **Covering**



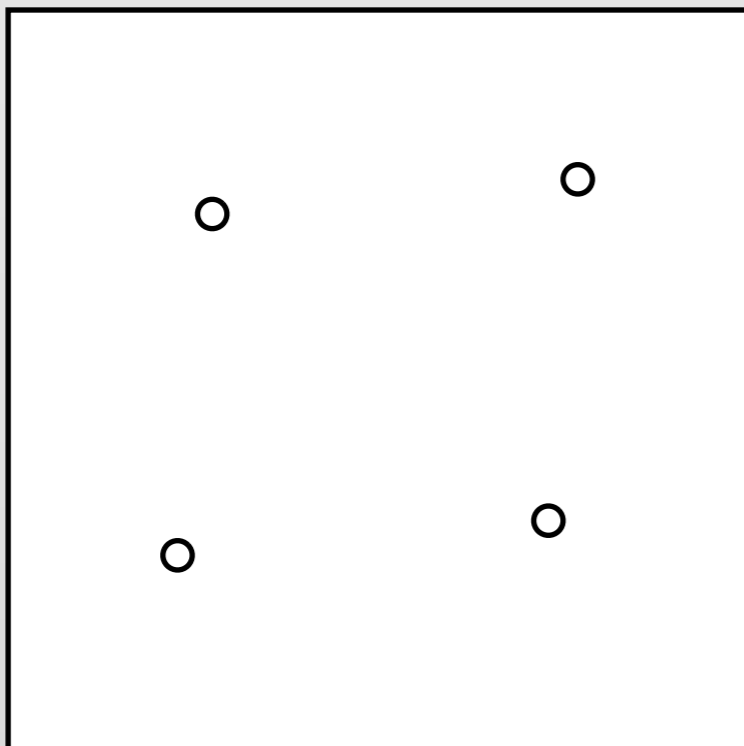
Example: Bounded Euclidean Space

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$ ← **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$ ← **Covering**



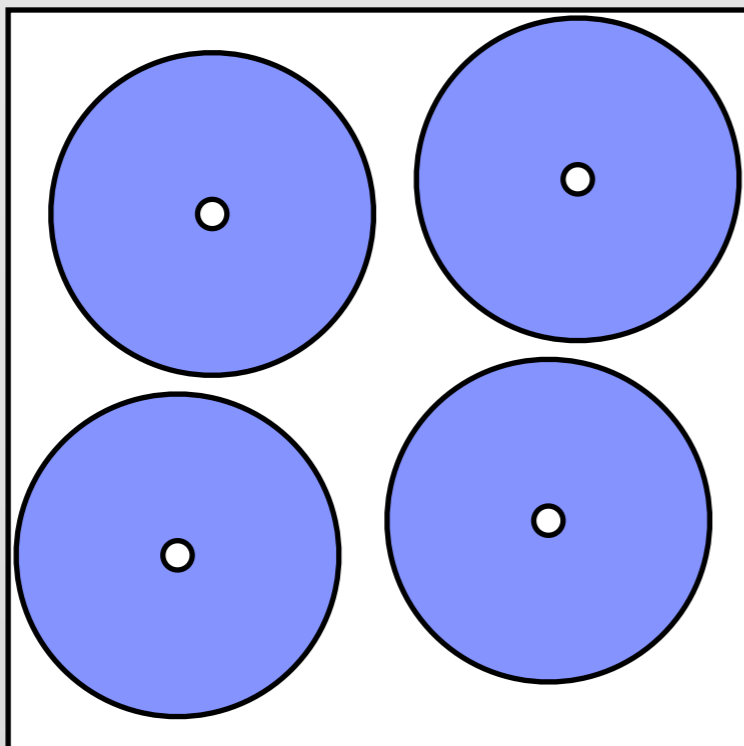
Example: Bounded Euclidean Space
Related to ball packing of radius $\varepsilon/2$.

“Nets catch everything that’s big.”

Definition 1. *Given a metric space X , a metric ε -net is a set $N \subset X$ such that*

1. $x, y \in N \Rightarrow \mathbf{d}(x, y) \geq \varepsilon$ ← **Packing**

2. $x \in X \Rightarrow \exists y \in N : \mathbf{d}(x, y) \leq \varepsilon$ ← **Covering**



Example: Bounded Euclidean Space
Related to ball packing of radius $\varepsilon/2$.

Meshes as Metric Nets

Meshes as Metric Nets

The underlying metric is not Euclidean (but it's close).

Meshes as Metric Nets

The underlying metric is not Euclidean (but it's close).

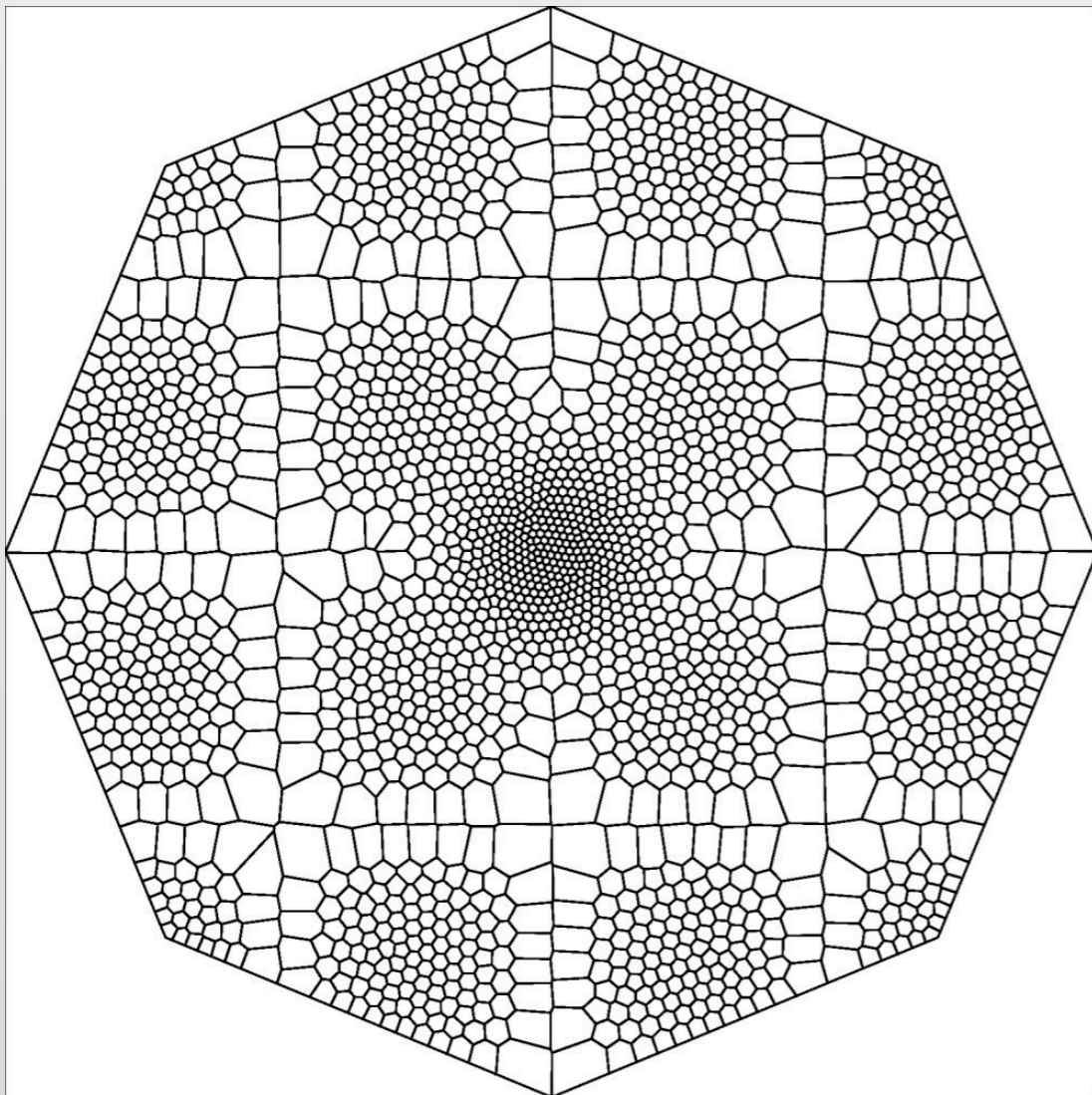


image source: Alice Project, INRIA

Meshes as Metric Nets

The underlying metric is not Euclidean (but it's close).

Metric is scaled according to the "feature size".

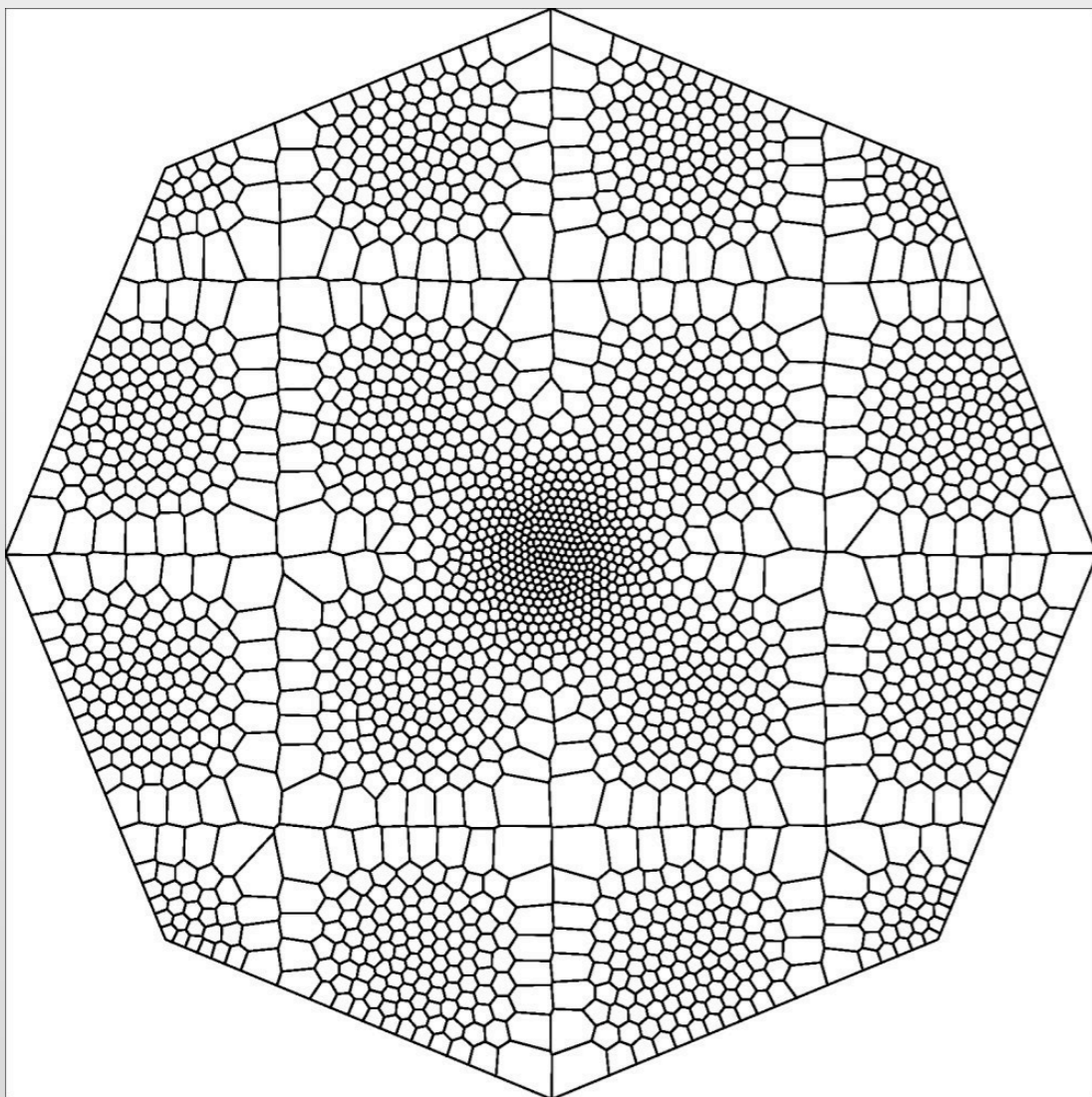


image source: Alice Project, INRIA

Meshes as Metric Nets

The underlying metric is not Euclidean (but it's close).

Metric is scaled according to the "feature size".

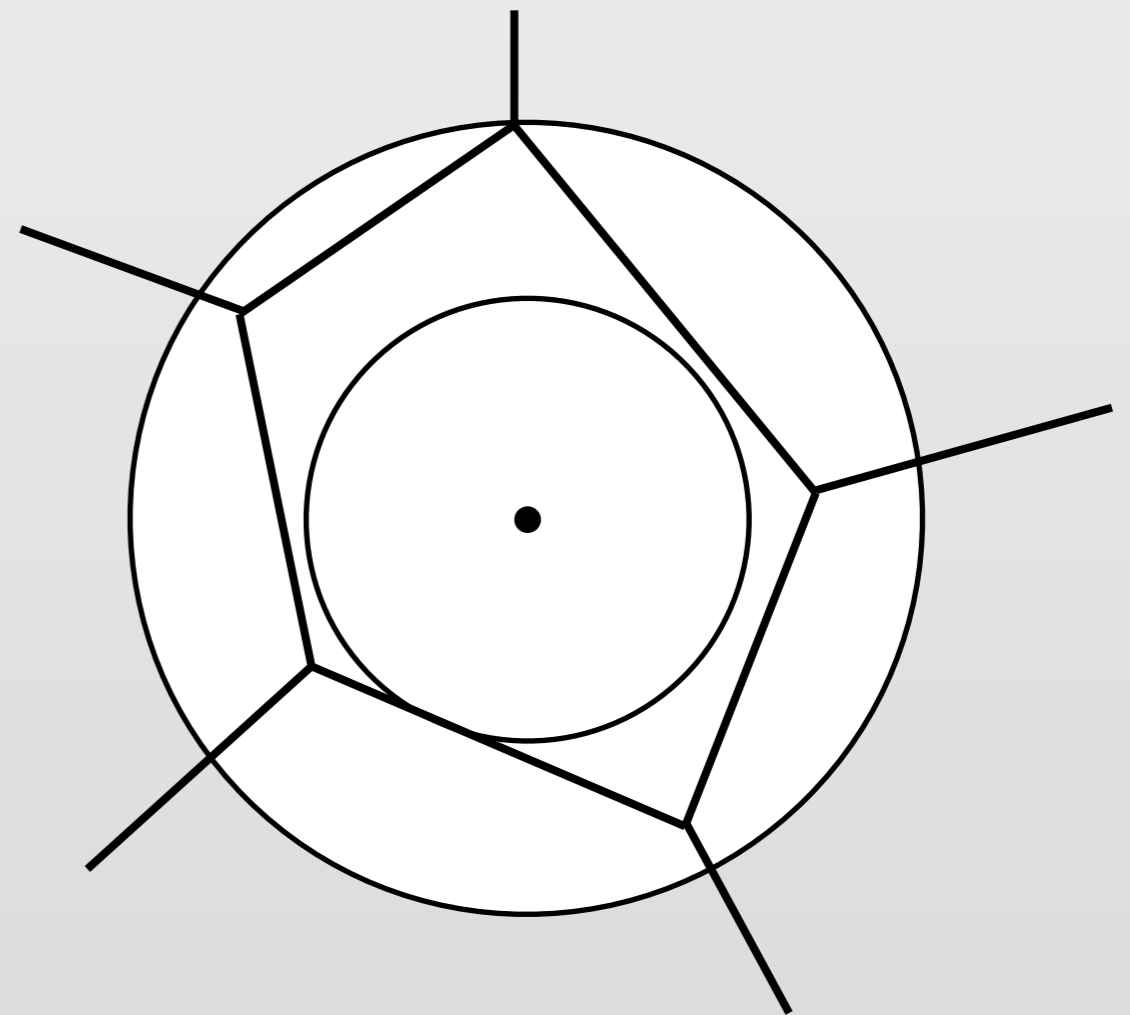
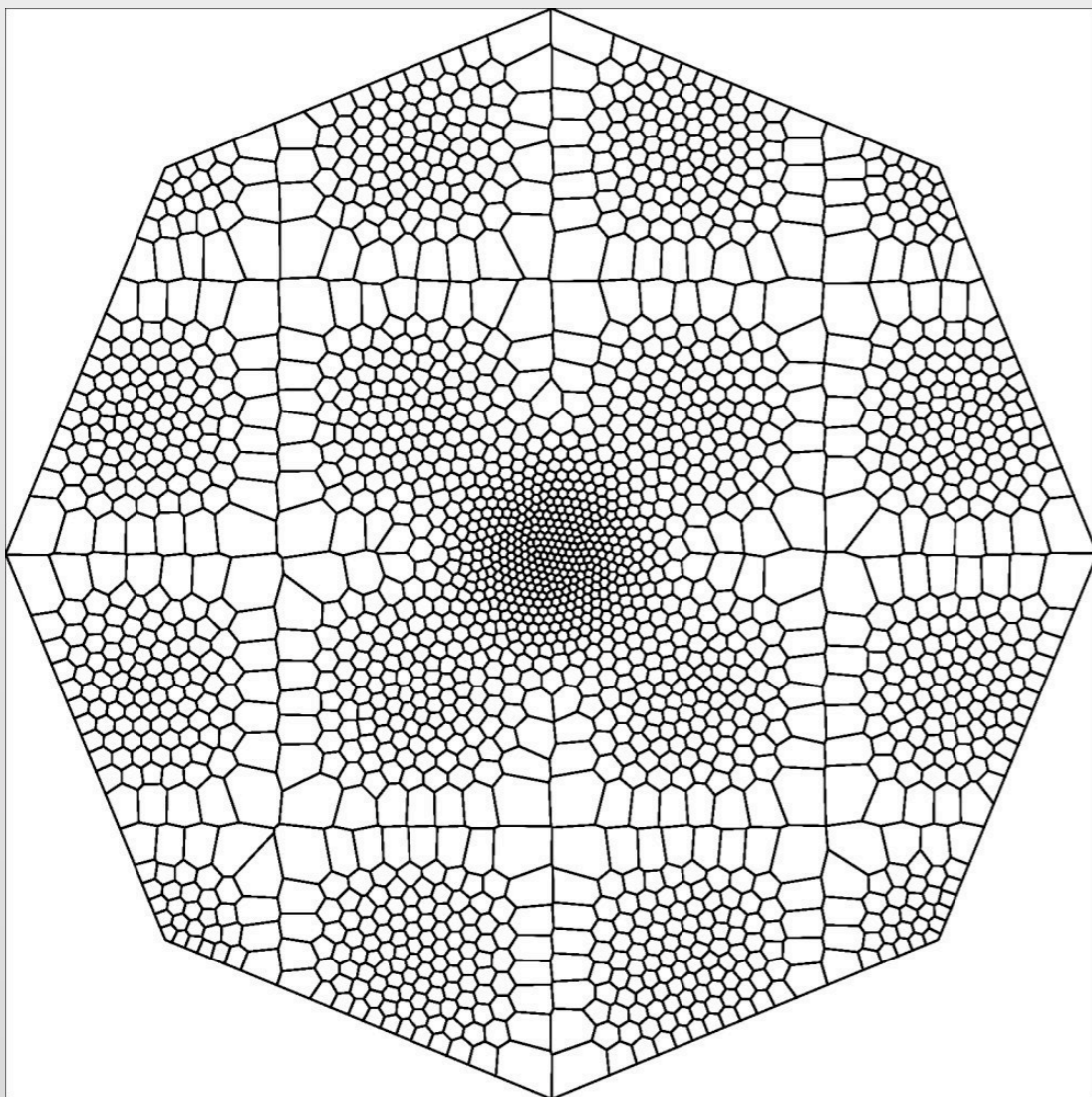


image source: Alice Project, INRIA

Meshes as Metric Nets

The underlying metric is not Euclidean (but it's close).

Metric is scaled according to the "feature size".

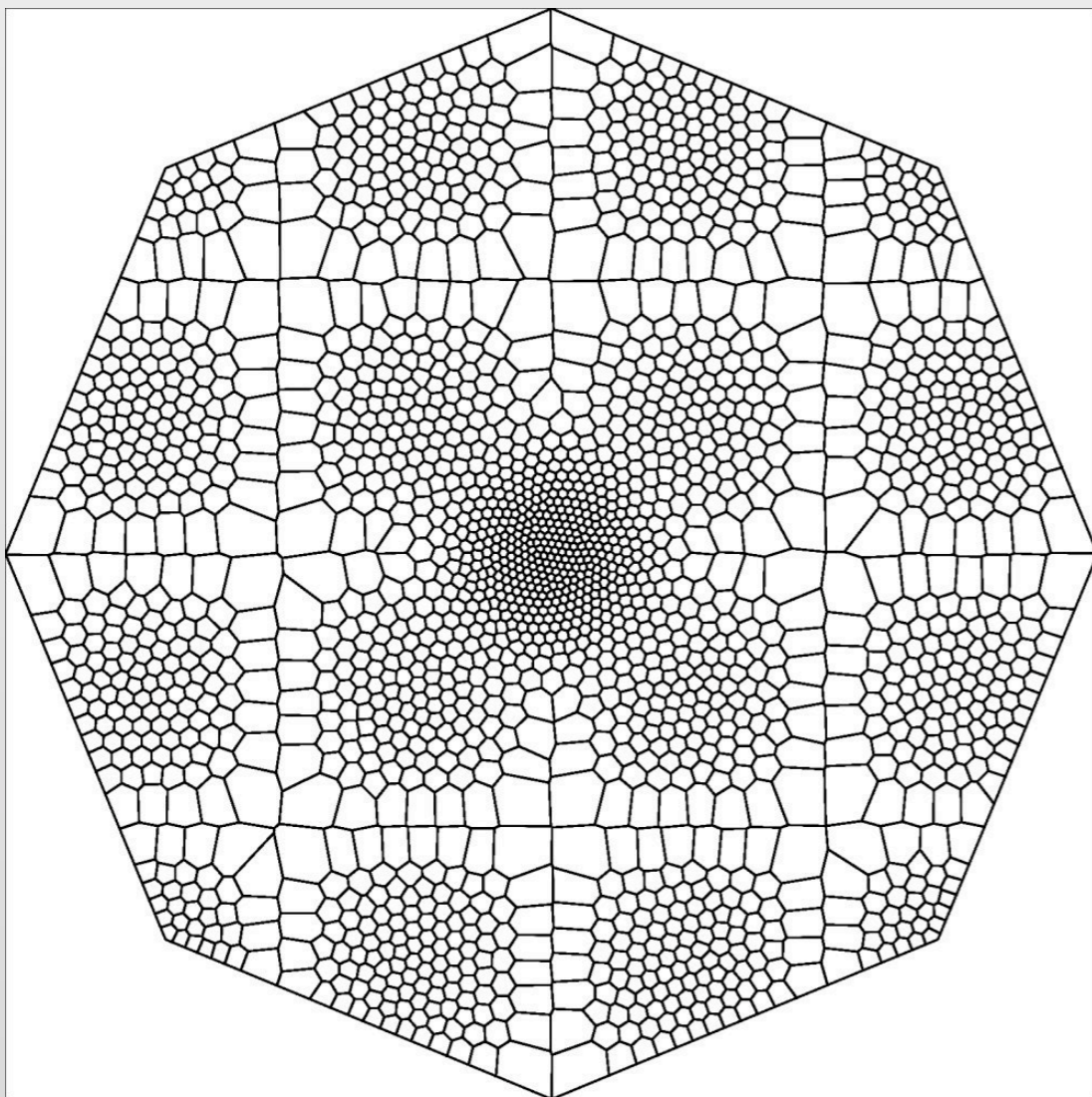
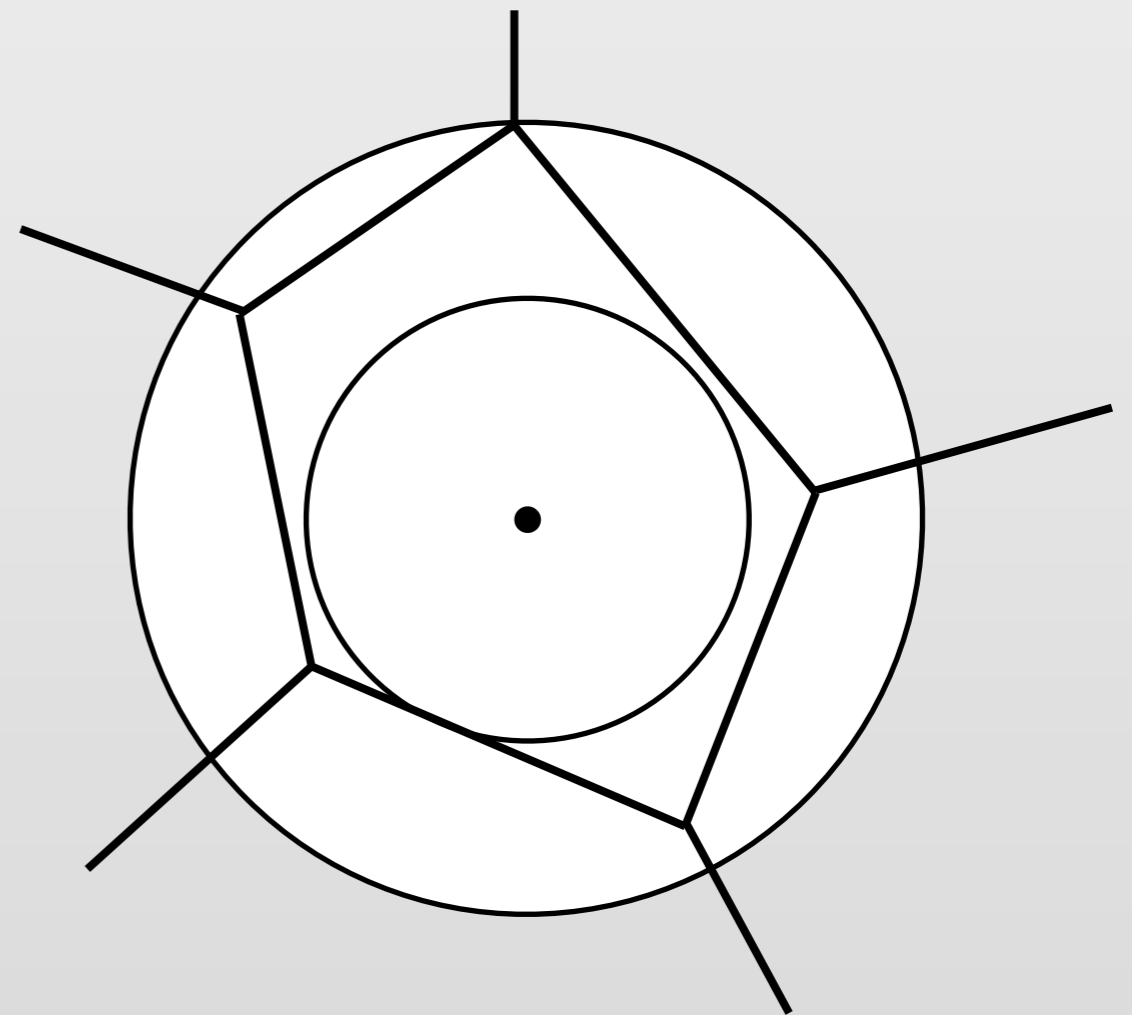


image source: Alice Project, INRIA



Quality = bounded aspect ratio

The Greedy Net Algorithm and SVR

The Greedy Net Algorithm and SVR

Add the **farthest** point from the current set.
Repeat.

The Greedy Net Algorithm and SVR

Add the **farthest** point from the current set.
Repeat.

Harder for infinite metric spaces.

- Figure out the metric
- Decide input or Steiner
- Do point location work

The Greedy Net Algorithm and SVR

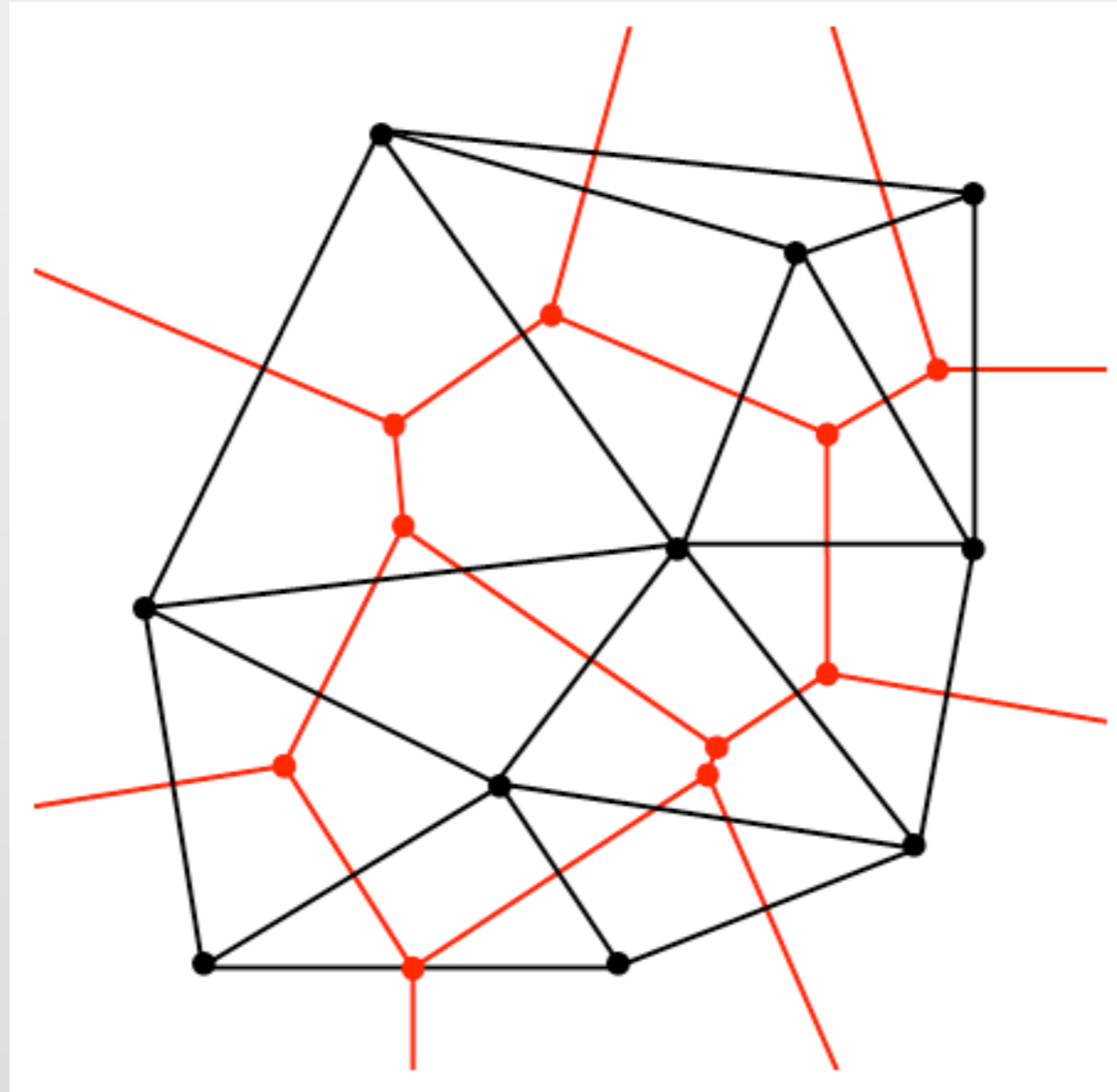
Add the **farthest** point from the current set.
Repeat.

Harder for infinite metric spaces.

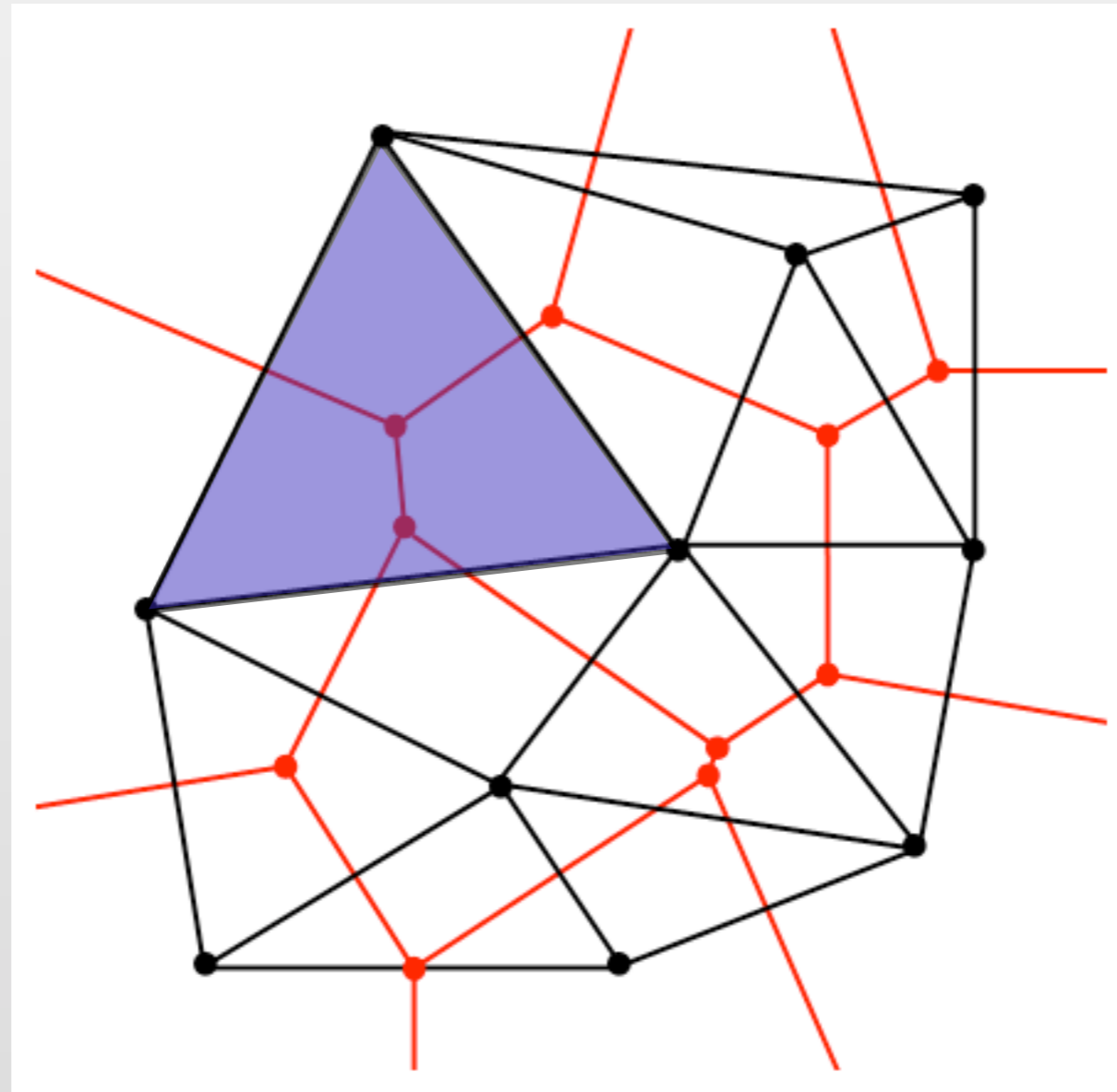
- Figure out the metric
- Decide input or Steiner
- Do point location work

[Sparse Voronoi Refinement](#) [Hudson, Miller, Phillips 2006]

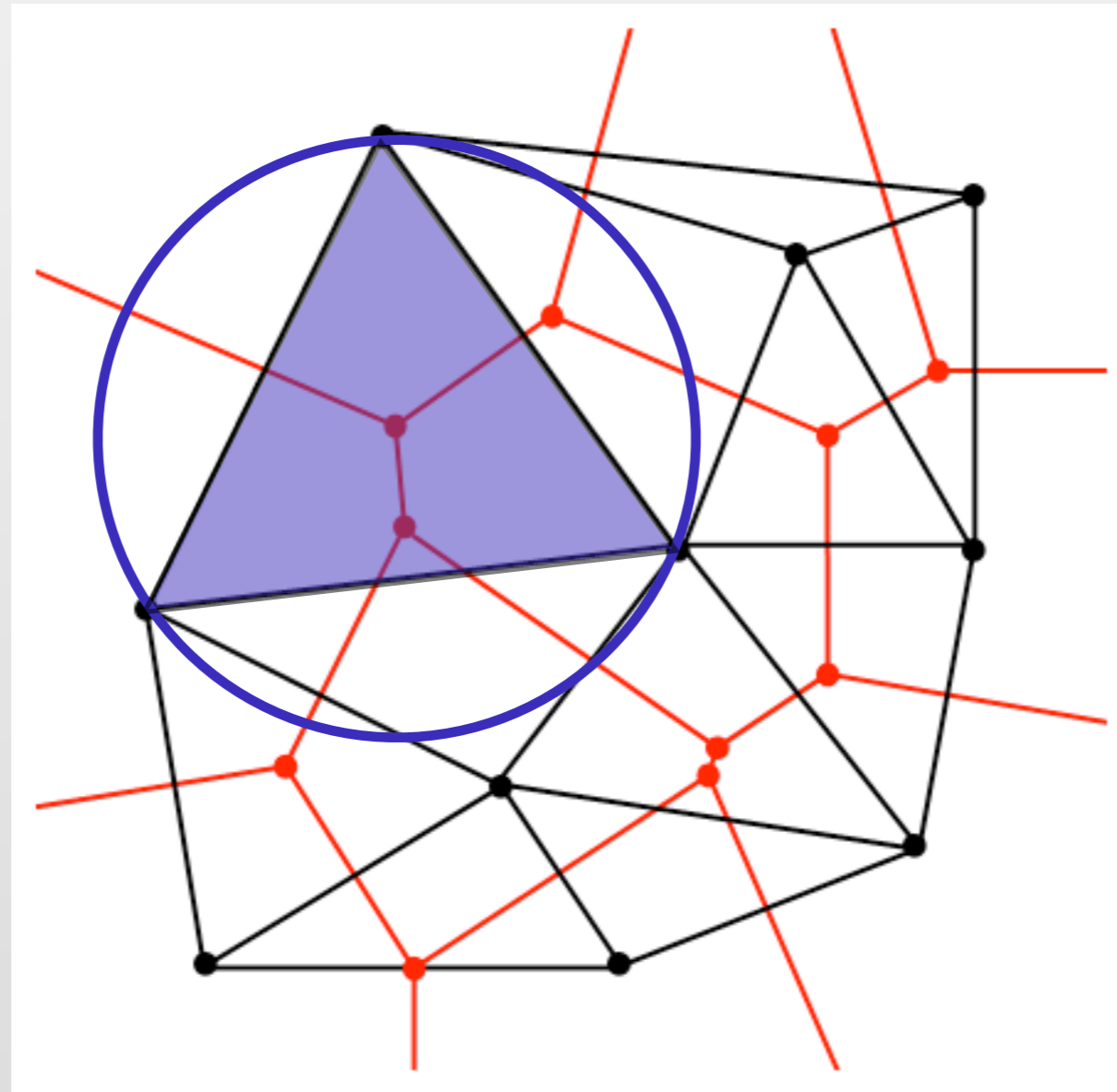
The Delaunay Triangulation is the dual of the Voronoi Diagram. 8



The Delaunay Triangulation is the dual of the Voronoi Diagram. 8

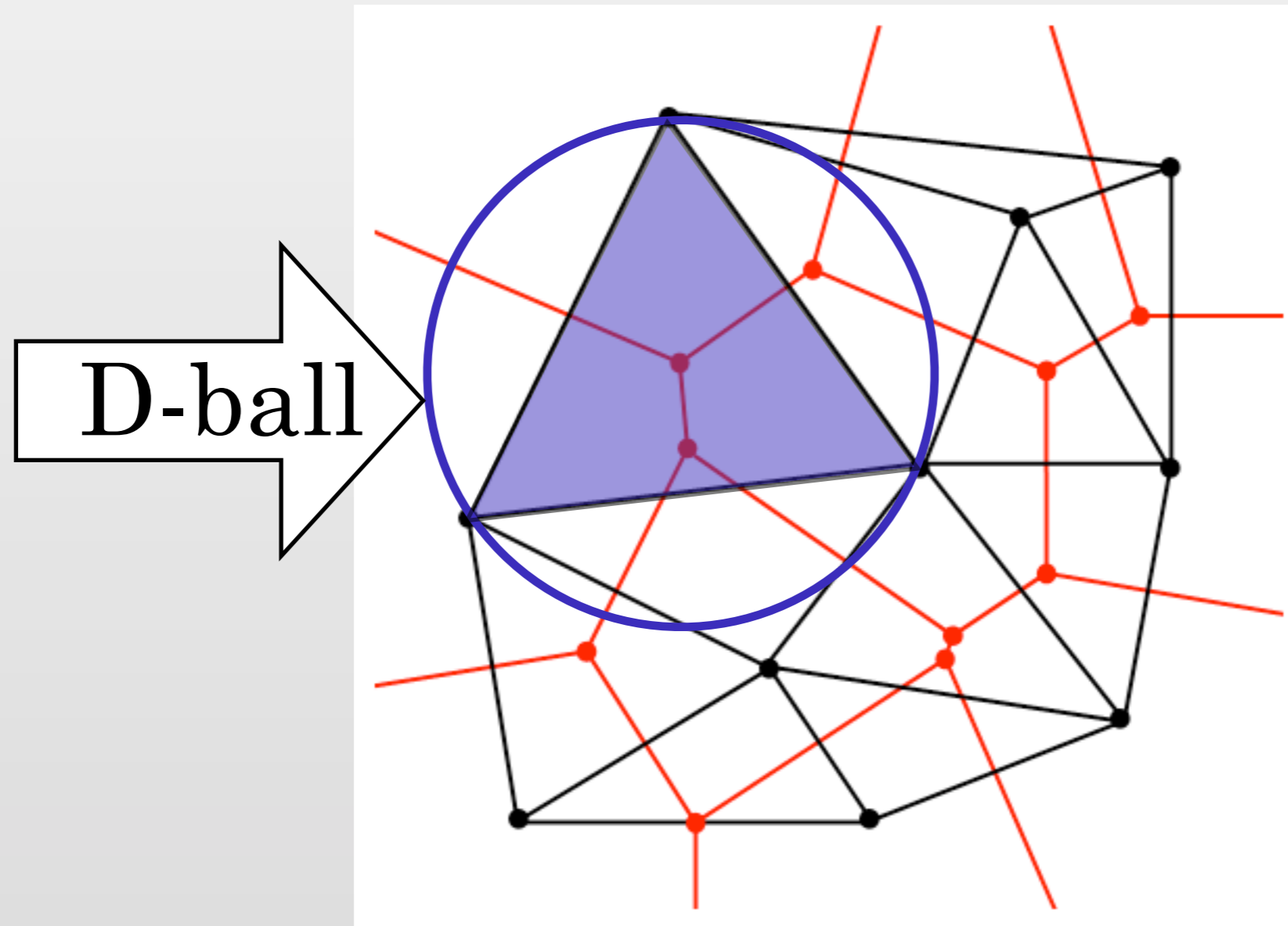


The Delaunay Triangulation is the dual of the Voronoi Diagram. 8

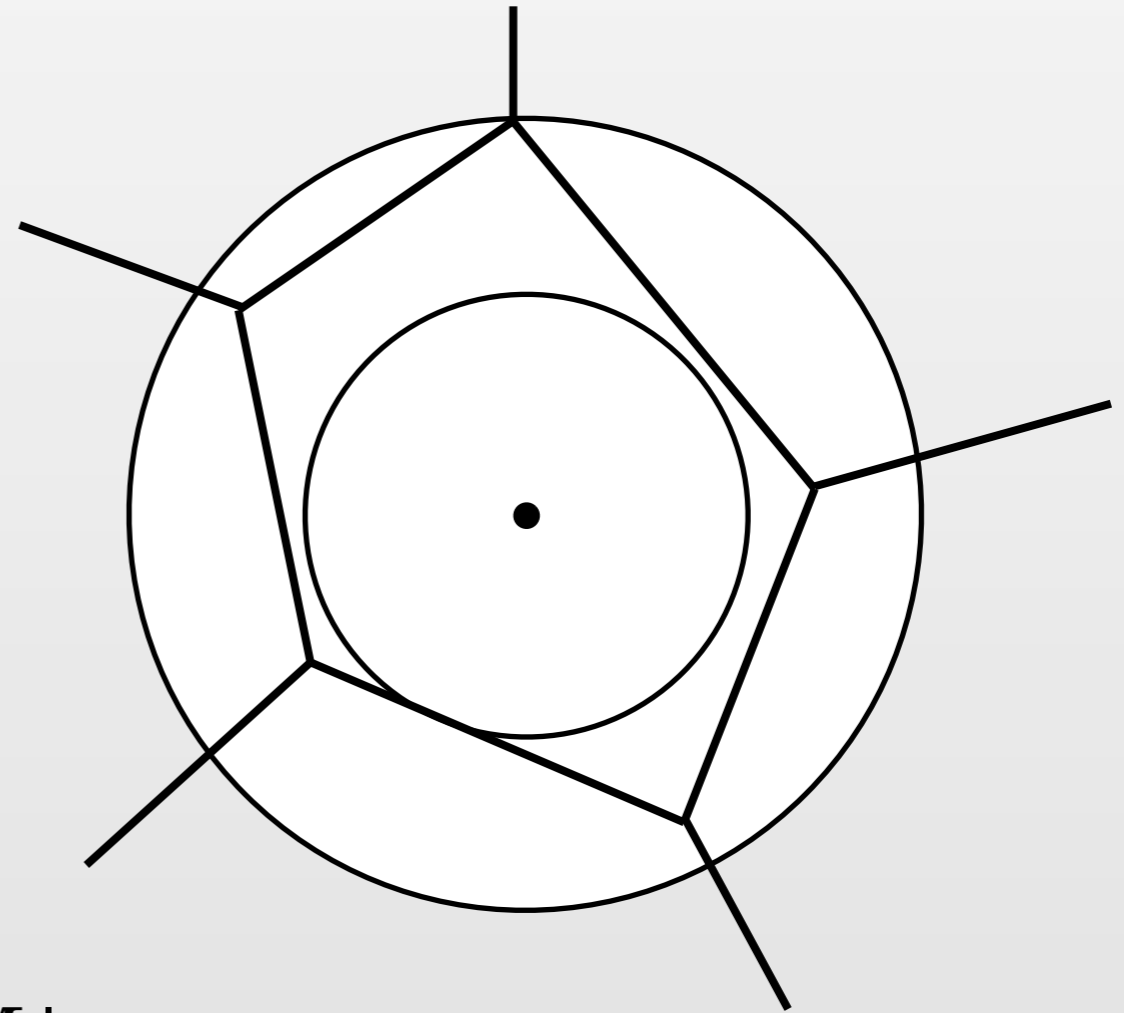


The Delaunay Triangulation is the dual
of the Voronoi Diagram.

8



Quality Meshes

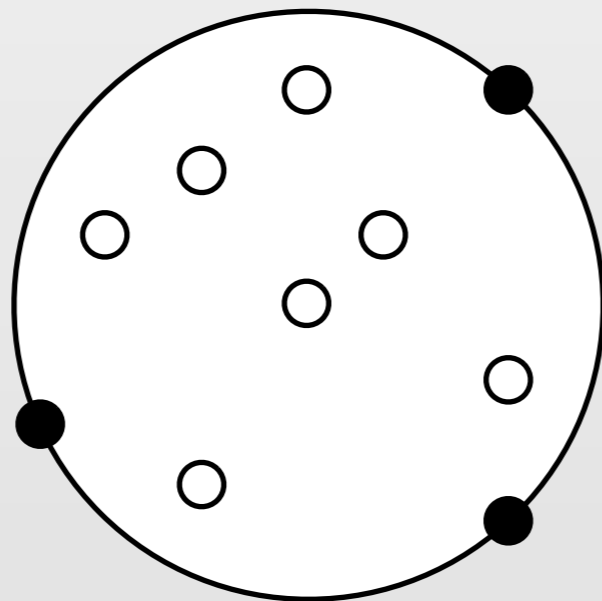


D-Balls are Constant Ply

Total Complexity is linear in $|M|$

No D-ball intersects more than $O(1)$ others.

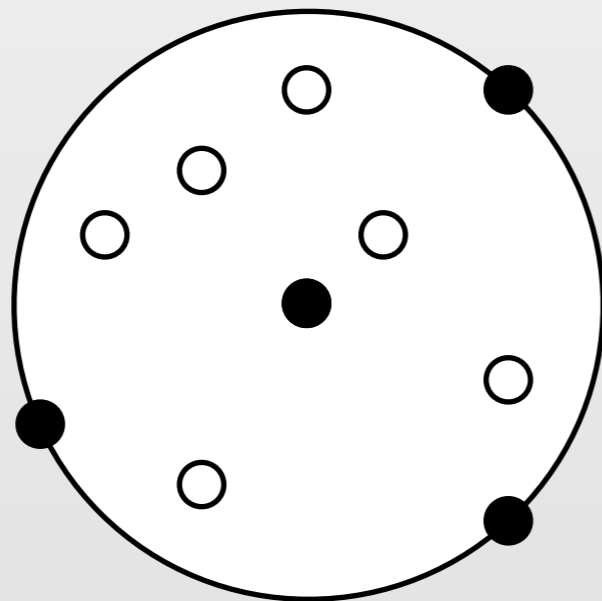
Idea: Store the uninserted points in the D-balls.
When the balls change, make local updates.



Lemma (HMP06). *A point is touched at most a constant number of times before the radius of the largest D-ball containing it goes down by a factor of 2.*

Geometric Divide and Conquer: $O(n \log \Delta)$

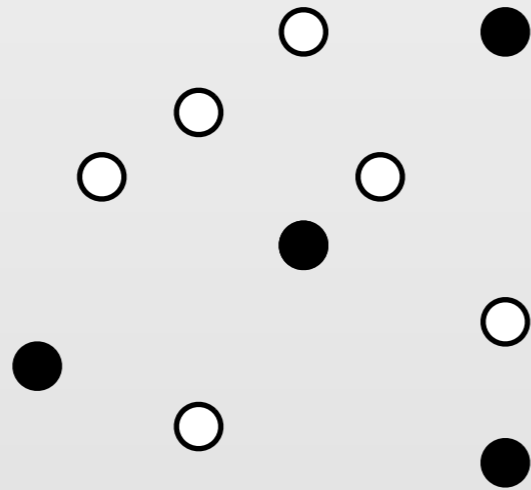
Idea: Store the uninserted points in the D-balls.
When the balls change, make local updates.



Lemma (HMP06). *A point is touched at most a constant number of times before the radius of the largest D-ball containing it goes down by a factor of 2.*

Geometric Divide and Conquer: $O(n \log \Delta)$

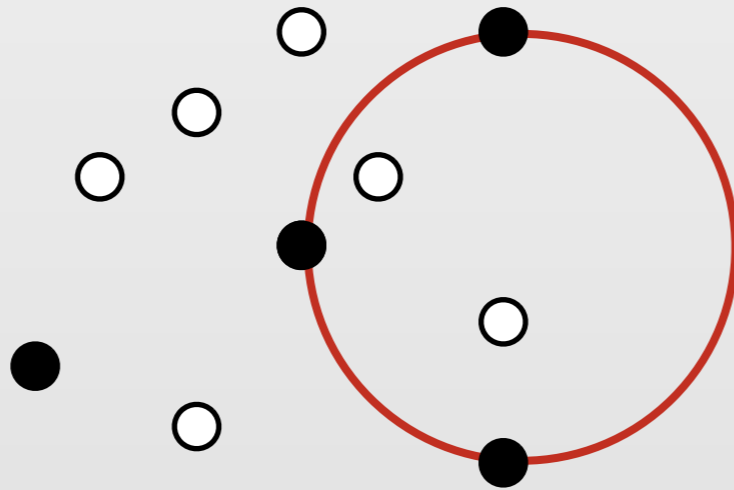
Idea: Store the uninserted points in the D-balls.
When the balls change, make local updates.



Lemma (HMP06). *A point is touched at most a constant number of times before the radius of the largest D-ball containing it goes down by a factor of 2.*

Geometric Divide and Conquer: $O(n \log \Delta)$

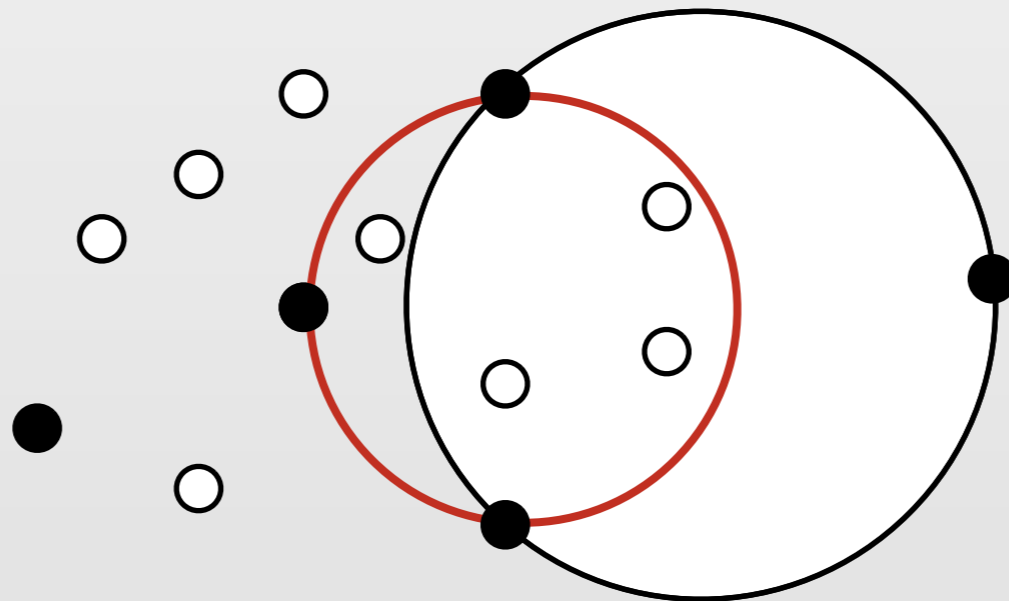
Idea: Store the uninserted points in the D-balls.
When the balls change, make local updates.



Lemma (HMP06). *A point is touched at most a constant number of times before the radius of the largest D-ball containing it goes down by a factor of 2.*

Geometric Divide and Conquer: $O(n \log \Delta)$

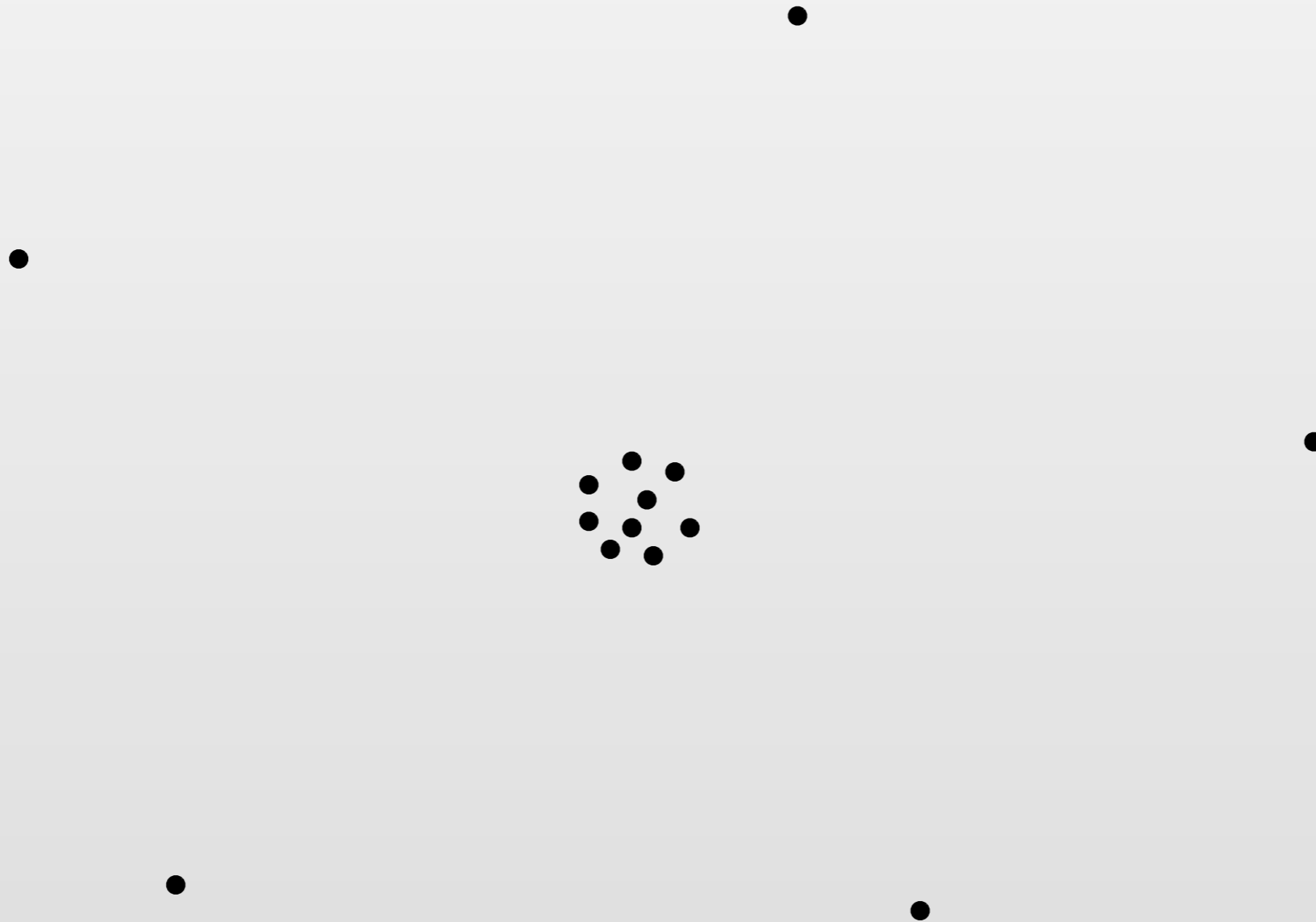
Idea: Store the uninserted points in the D-balls.
When the balls change, make local updates.



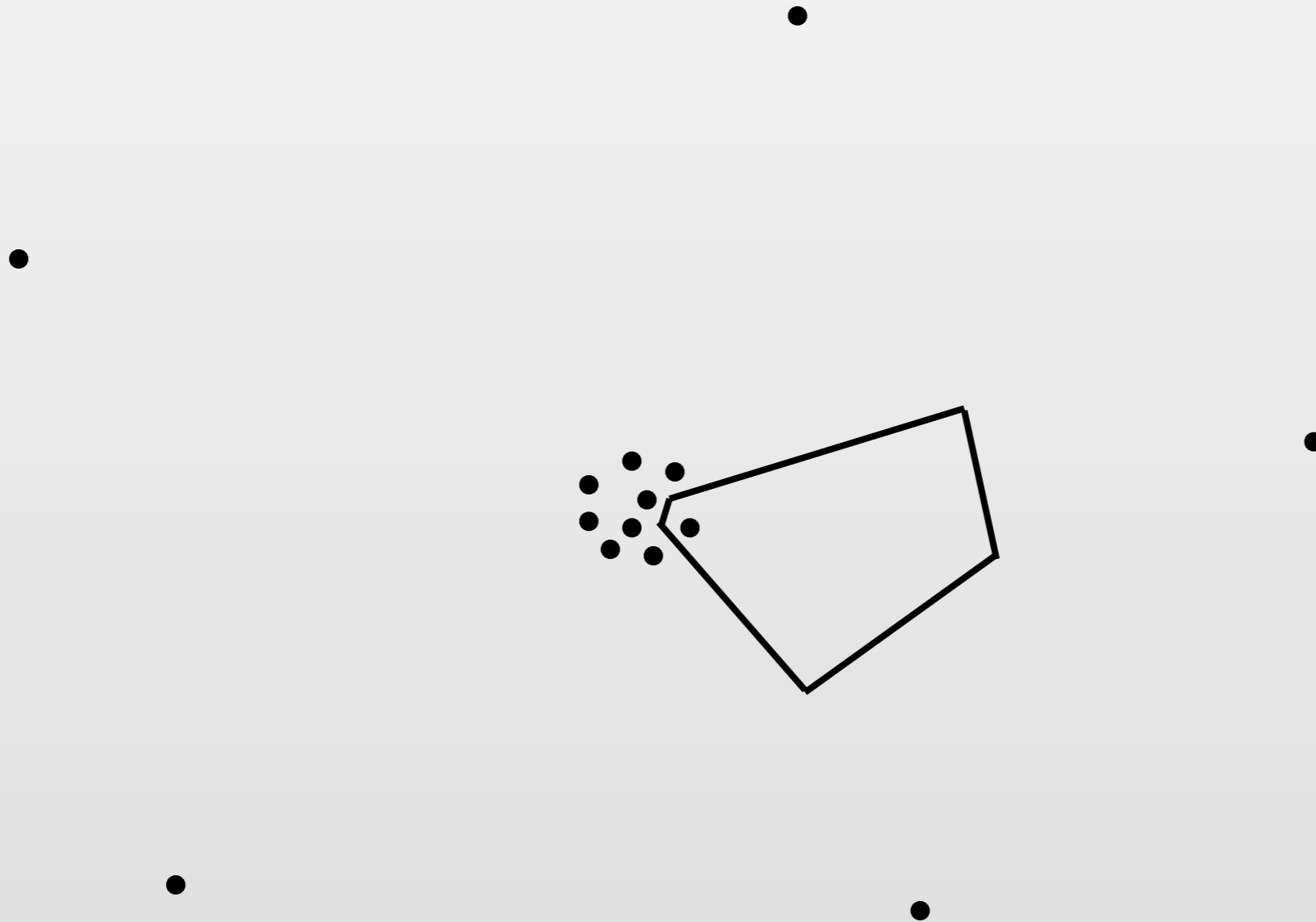
Lemma (HMP06). *A point is touched at most a constant number of times before the radius of the largest D-ball containing it goes down by a factor of 2.*

Geometric Divide and Conquer: $O(n \log \Delta)$

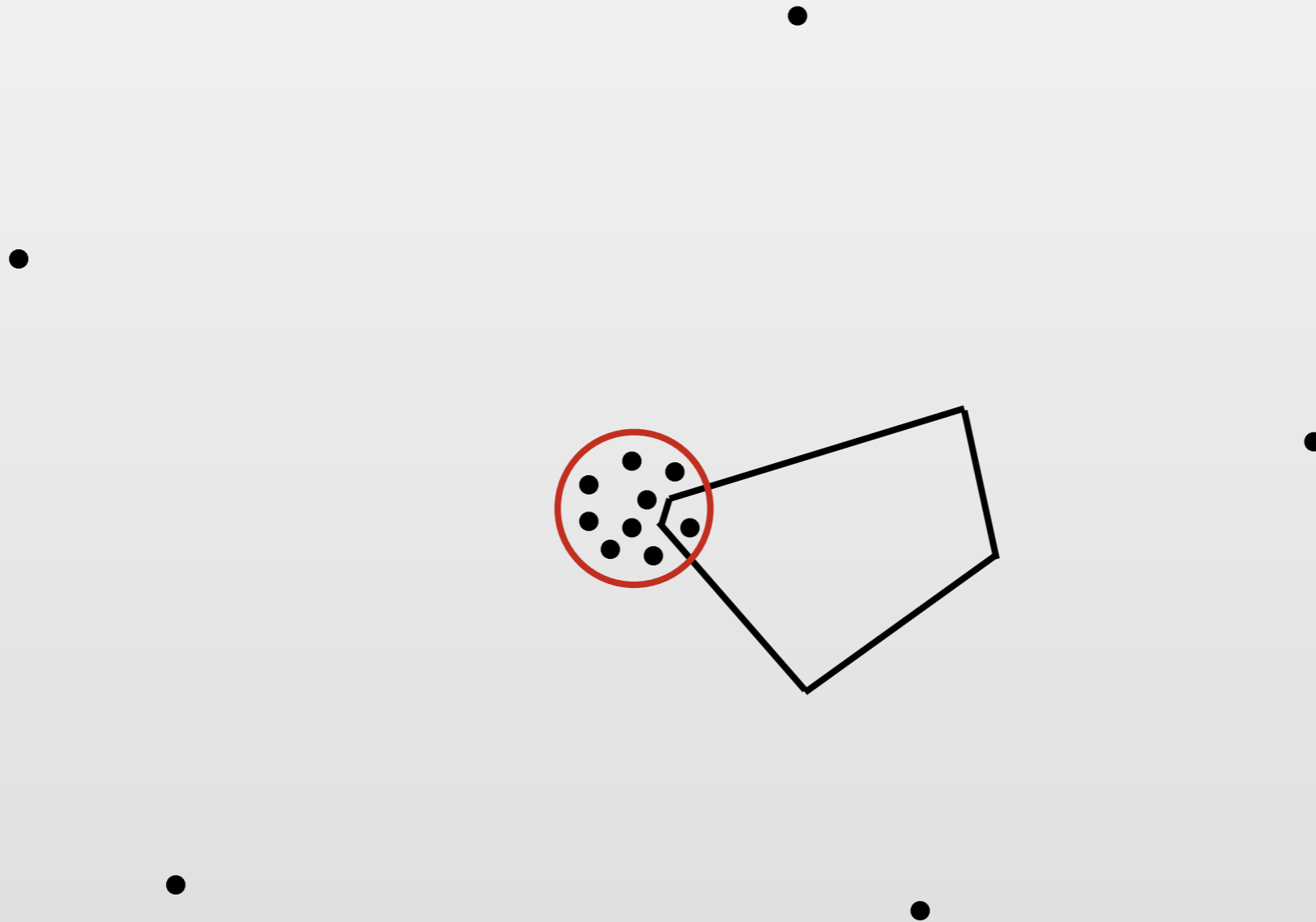
We replace *quality* with *hierarchical quality*.



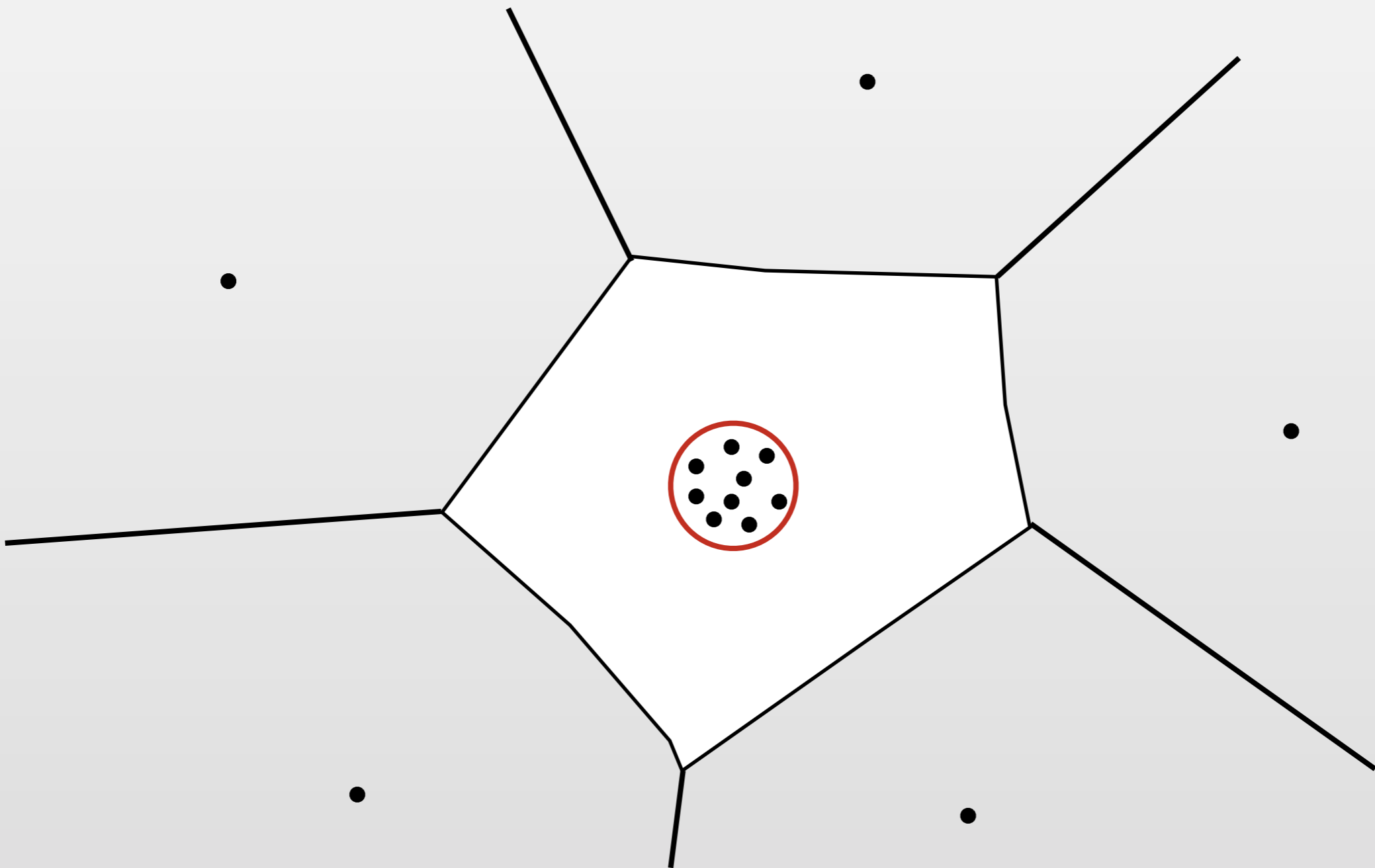
We replace *quality* with *hierarchical quality*.



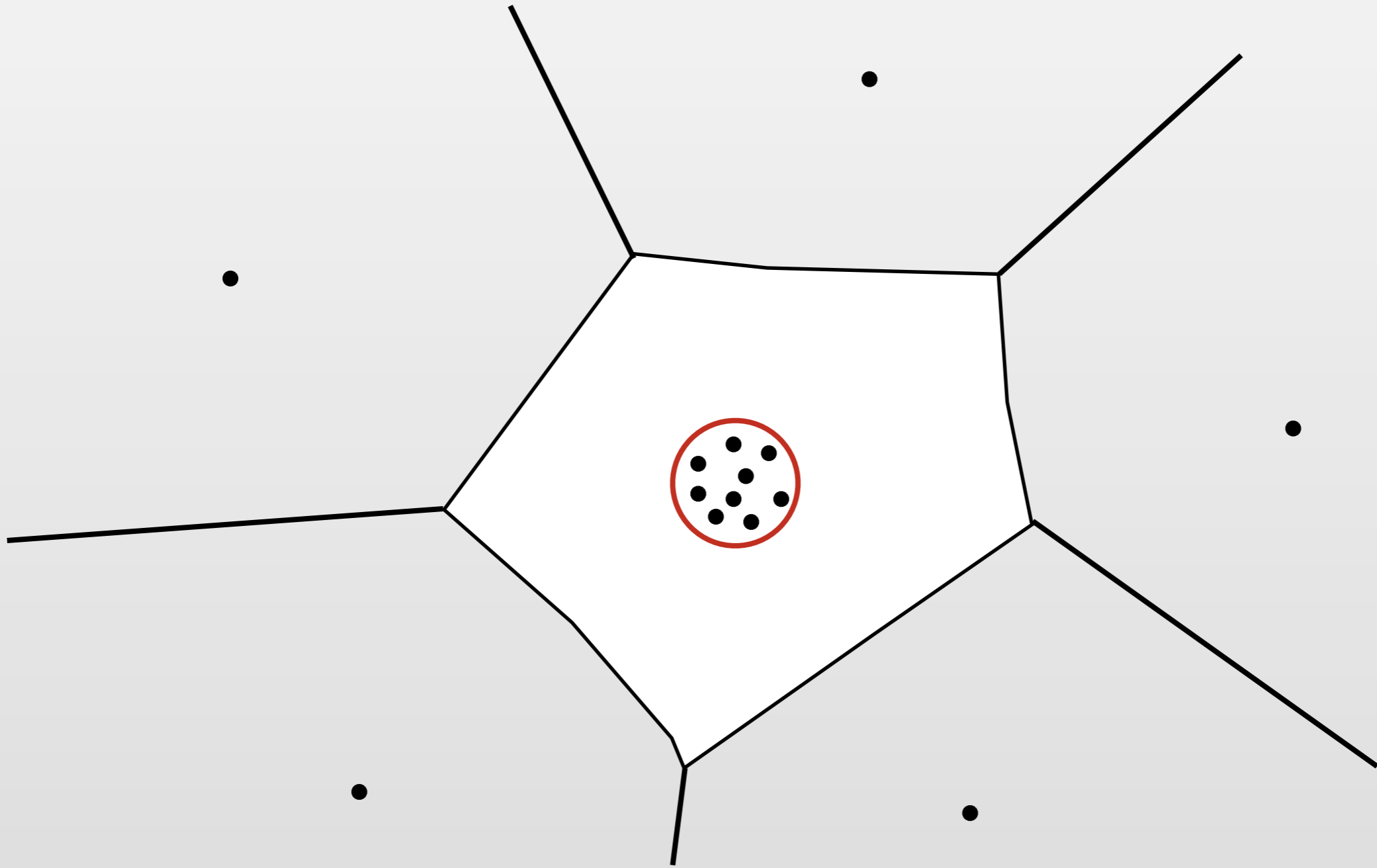
We replace *quality* with *hierarchical quality*.



We replace *quality* with *hierarchical quality*.

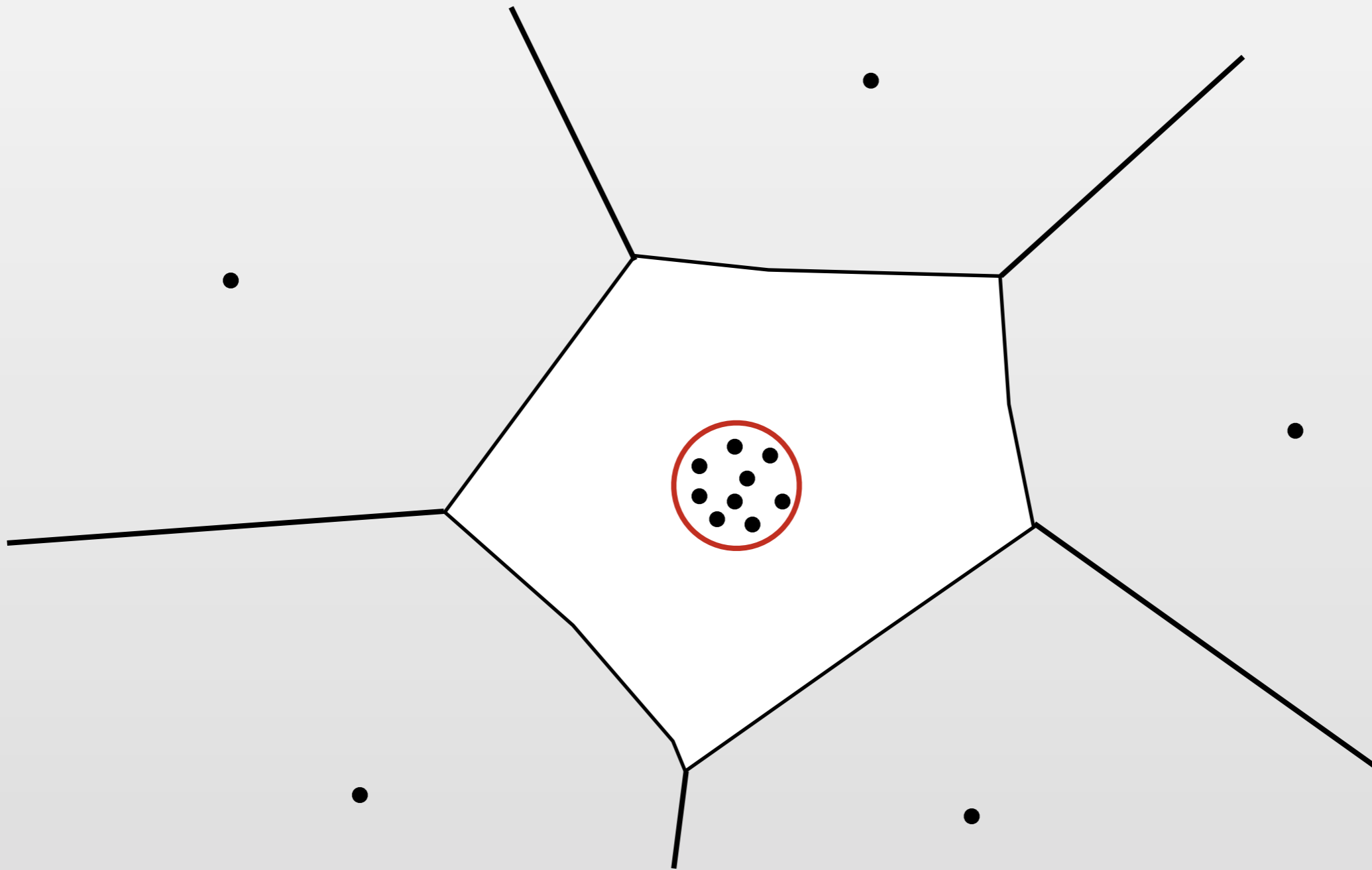


We replace *quality* with *hierarchical quality*.



Inside the cage: Old definition of quality.

We replace *quality* with *hierarchical quality*.

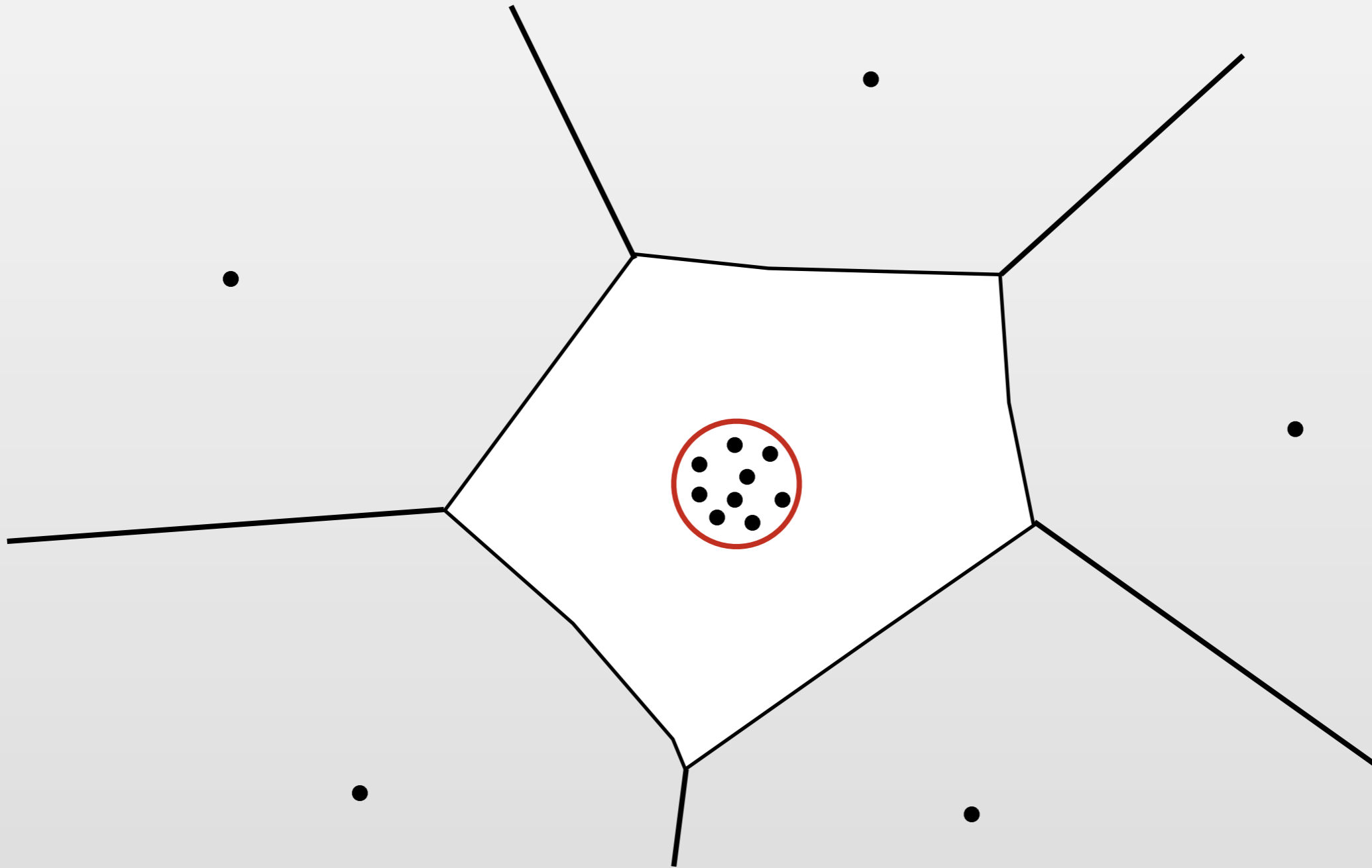


Inside the cage: Old definition of quality.

Outside: Treat the whole cage as a single object.

We replace *quality* with *hierarchical quality*.

11



Inside the cage: Old definition of quality.

Outside: Treat the whole cage as a single object.

All the same properties as quality meshes: ply, degree, etc.

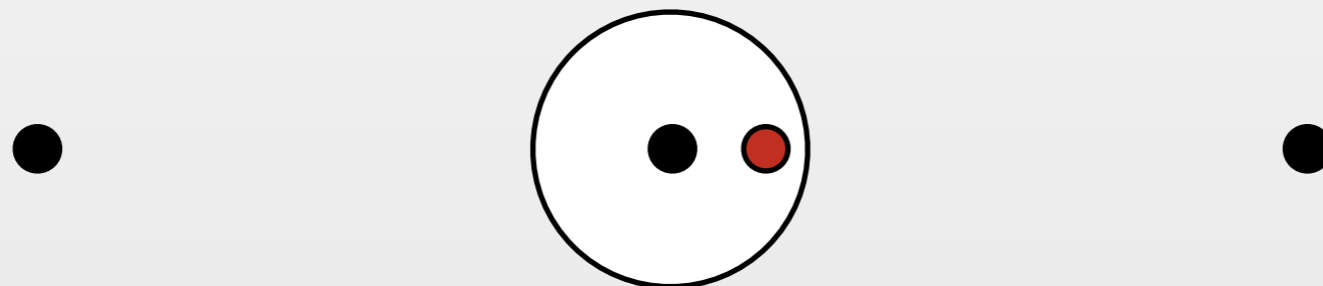
With hierarchical meshes, we can add inputs
in any order!



With hierarchical meshes, we can add inputs
in any order!

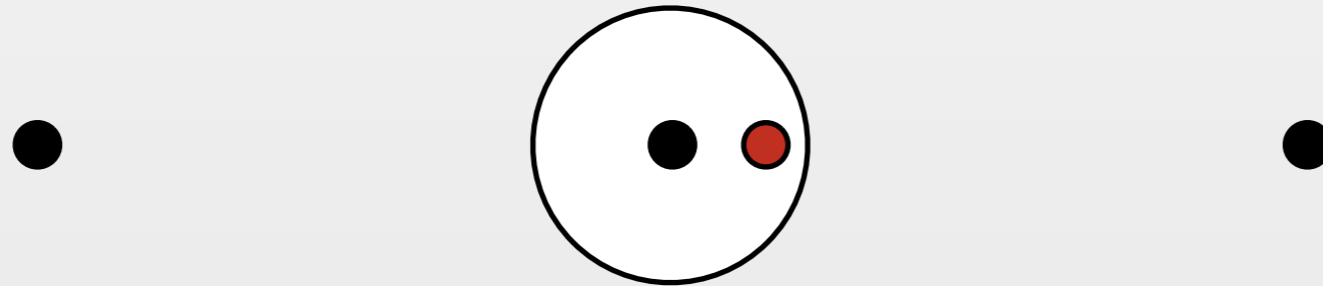


With hierarchical meshes, we can add inputs
in any order!



With hierarchical meshes, we can add inputs
in any order!

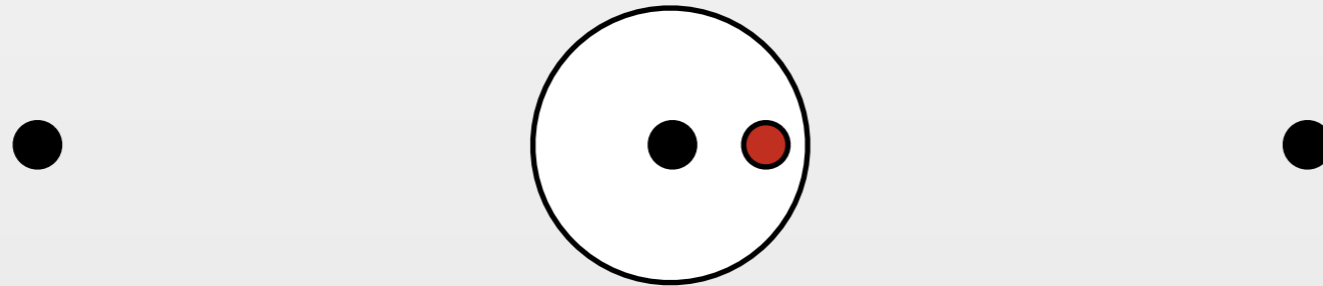
12



Goal: Combinatorial Divide and Conquer

With hierarchical meshes, we can add inputs in any order!

12

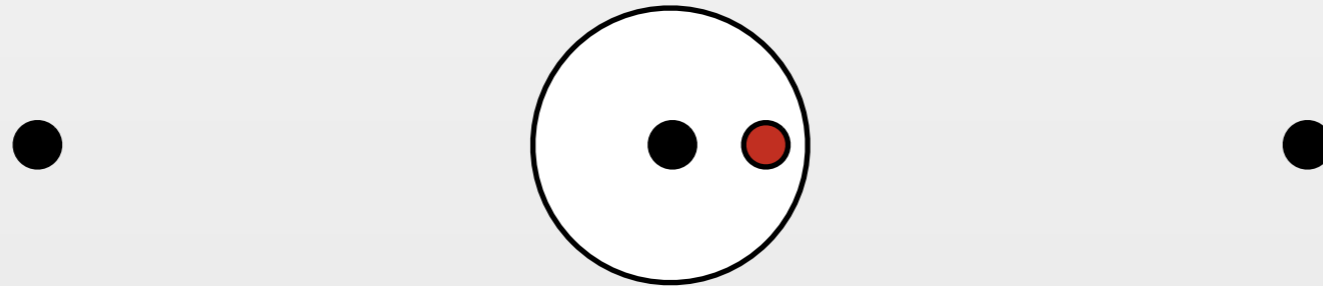


Goal: Combinatorial Divide and Conquer

Old: Progress was decreasing the radius by a factor of 2.

With hierarchical meshes, we can add inputs in any order!

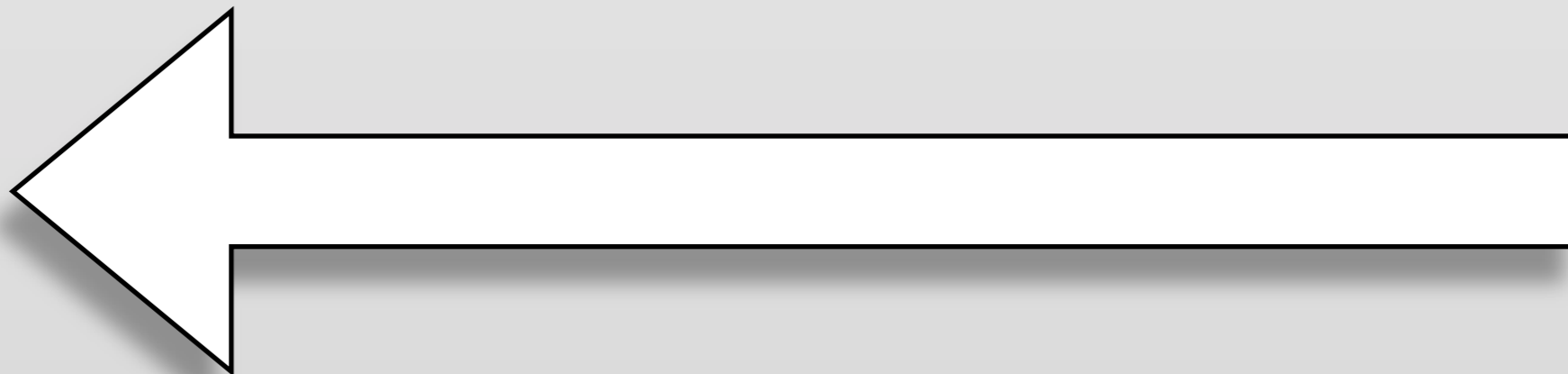
12



Goal: Combinatorial Divide and Conquer

Old: Progress was decreasing the radius by a factor of 2.

New: Decrease number of points in a ball by a factor of 2.



Range Nets

“Nets catch everything that’s big.”

“Nets catch everything that’s big.”

Definition. *A range space is a pair (X, R) , where X is a set (the vertices) and R is a collection of subsets (the ranges).*

“Nets catch everything that’s big.”

Definition. *A range space is a pair (X, R) , where X is a set (the vertices) and R is a collection of subsets (the ranges).*

For us $X = P$ and R is the set of open balls.

“Nets catch everything that’s big.”

Definition. *A range space is a pair (X, R) , where X is a set (the vertices) and R is a collection of subsets (the ranges).*

For us $X = P$ and R is the set of open balls.

Definition. *Given a range space (X, R) , a set $N \subset X$ is a **range space ε -net** if for all ranges $r \in R$ that contain at least $\varepsilon|X|$ vertices, r contains a vertex from N .*

“Nets catch everything that’s big.”

Definition. *A range space is a pair (X, R) , where X is a set (the vertices) and R is a collection of subsets (the ranges).*

For us $X = P$ and R is the set of open balls.

Definition. *Given a range space (X, R) , a set $N \subset X$ is a **range space ε -net** if for all ranges $r \in R$ that contain at least $\varepsilon|X|$ vertices, r contains a vertex from N .*

Theorem: [Chazelle & Matousek 96] For ε, d fixed constants, ε -nets of size $O(1)$ can be computed in $O(n)$ deterministic time.

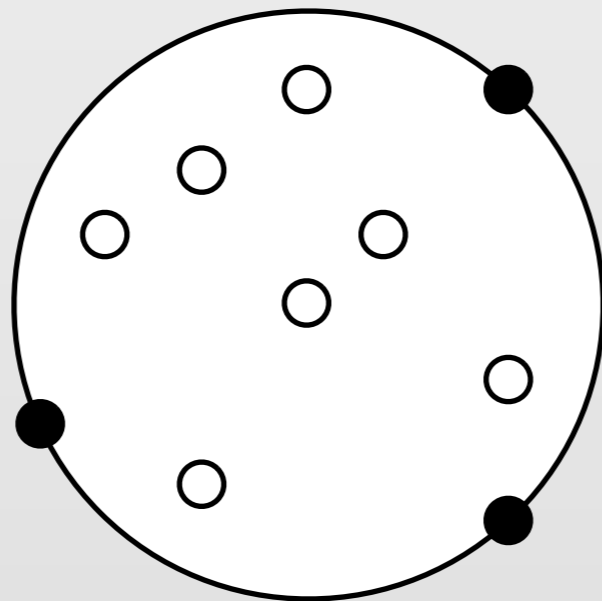
Ordering the inputs

For each D-Ball, select a $1/2d$ -net of the points it contains.

Take the union of these nets and call it a round.

Insert these.

Repeat.



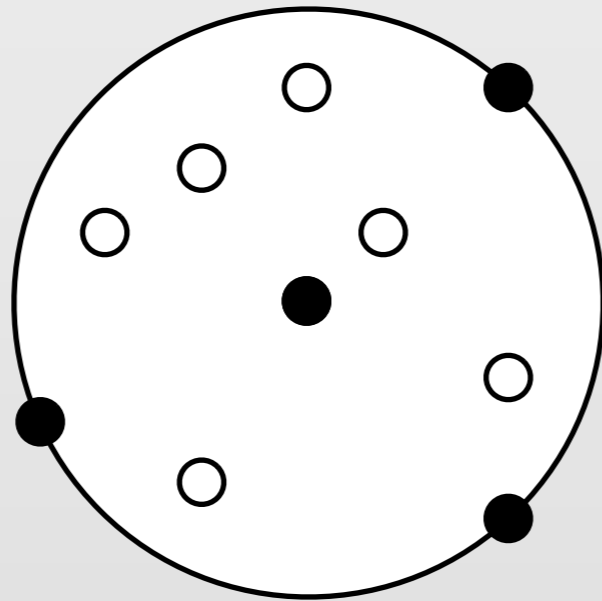
Ordering the inputs

For each D-Ball, select a $1/2d$ -net of the points it contains.

Take the union of these nets and call it a round.

Insert these.

Repeat.



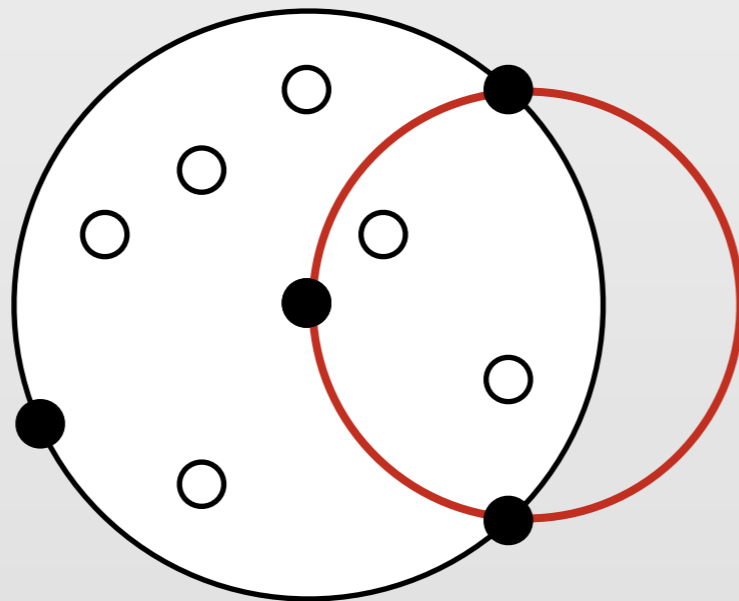
Ordering the inputs

For each D-Ball, select a $1/2d$ -net of the points it contains.

Take the union of these nets and call it a round.

Insert these.

Repeat.

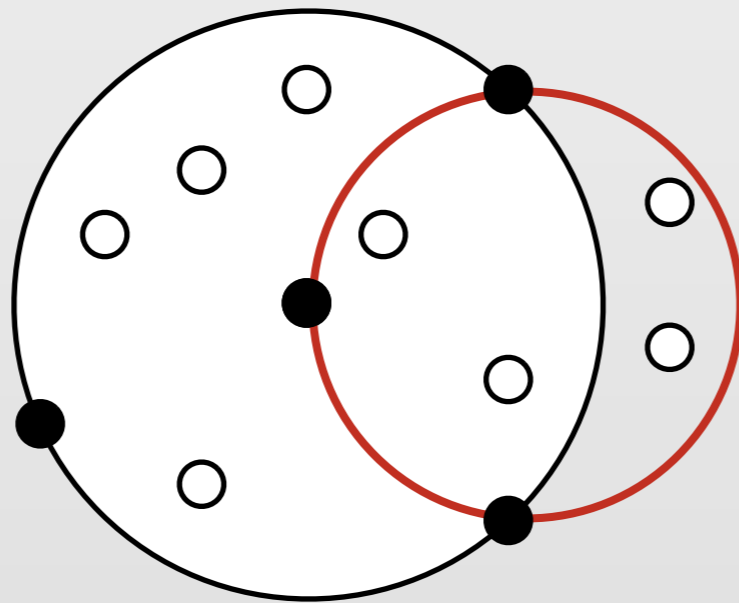


Ordering the inputs

For each D-Ball, select a $1/2d$ -net of the points it contains.
Take the union of these nets and call it a round.

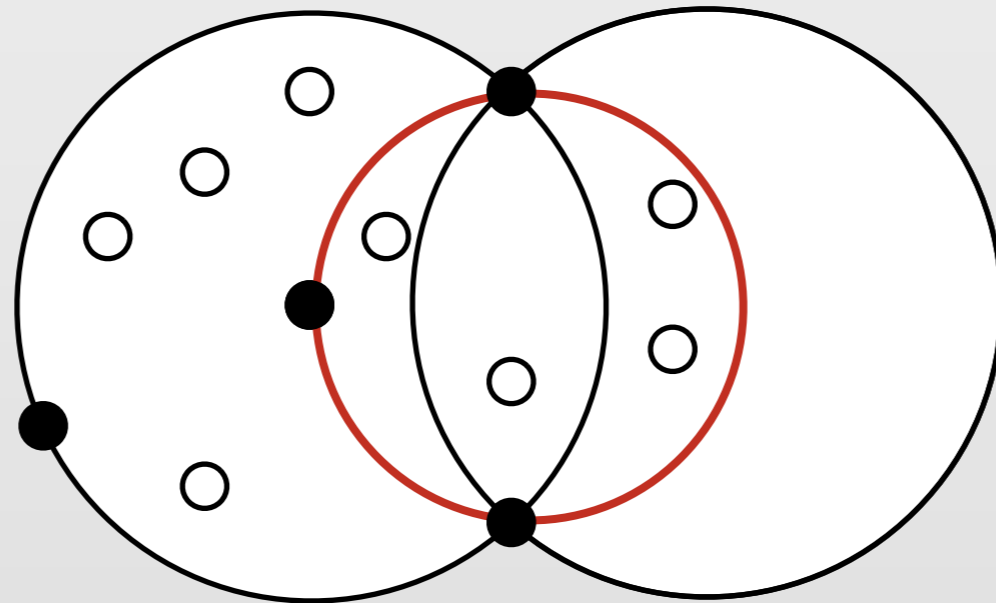
Insert these.

Repeat.



Ordering the inputs

For each D-Ball, select a $1/2d$ -net of the points it contains.
Take the union of these nets and call it a round.
Insert these.
Repeat.

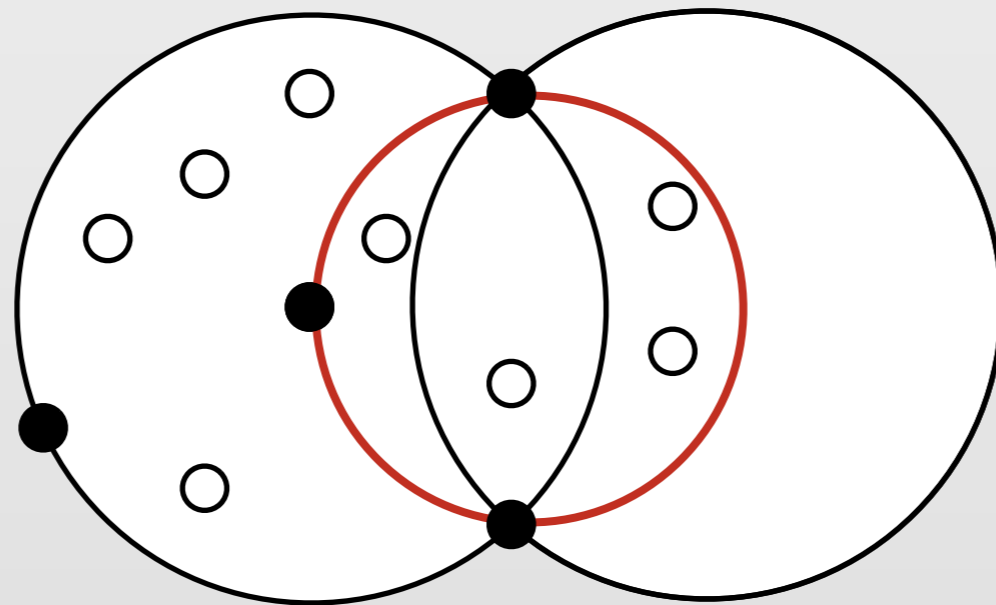


Ordering the inputs

For each D -Ball, select a $1/2d$ -net of the points it contains.
Take the union of these nets and call it a round.

Insert these.

Repeat.



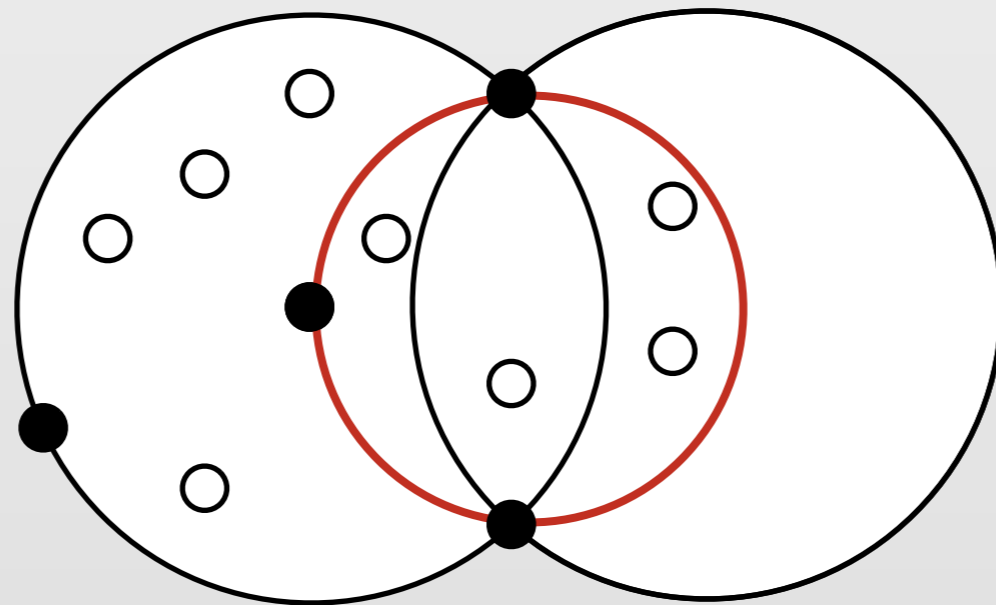
Lemma. *Let M be a set of vertices. If an open ball B contains no points of M , then B is contained in the union of d D -balls of M .*

Ordering the inputs

For each D -Ball, select a $1/2d$ -net of the points it contains.
Take the union of these nets and call it a round.

Insert these.

Repeat.



$\log(n)$ Rounds

Lemma. *Let M be a set of vertices. If an open ball B contains no points of M , then B is contained in the union of d D -balls of M .*

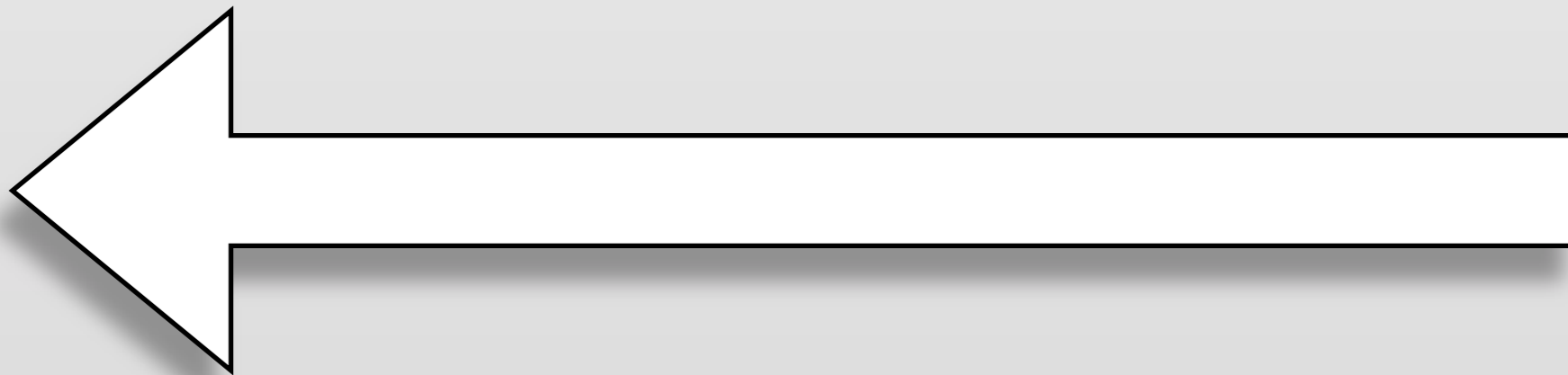
The D-Ball Covering Lemma

Lemma. *Let M be a set of vertices. If an open ball B contains no points of M , then B is contained in the union of d D-balls of M .*

The D-Ball Covering Lemma

Lemma. *Let M be a set of vertices. If an open ball B contains no points of M , then B is contained in the union of d D-balls of M .*

Proof is constructive using Carathéodory's Theorem.



Amortized Cost of a Round is $O(n)$

16

Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.

Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

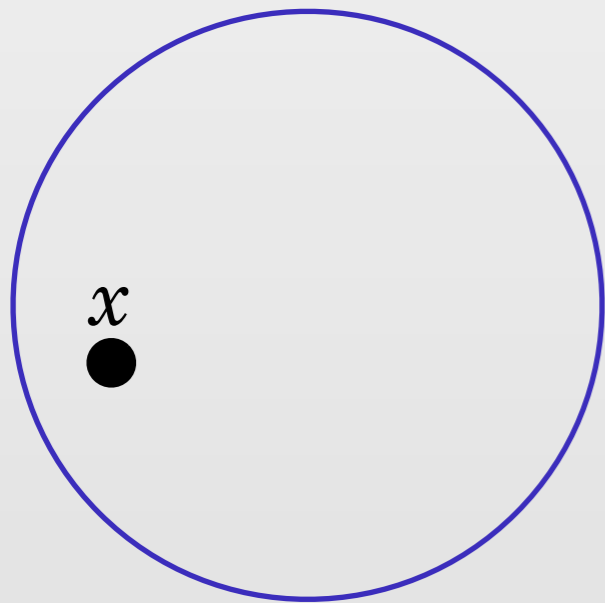
Claim: x only gets touched $O(1)$ times per round.

x
●

Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

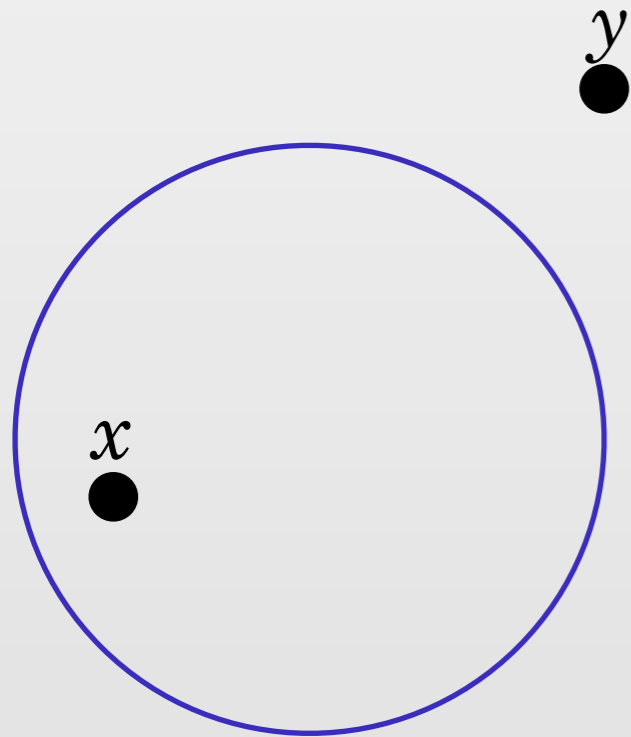
Claim: x only gets touched $O(1)$ times per round.



Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

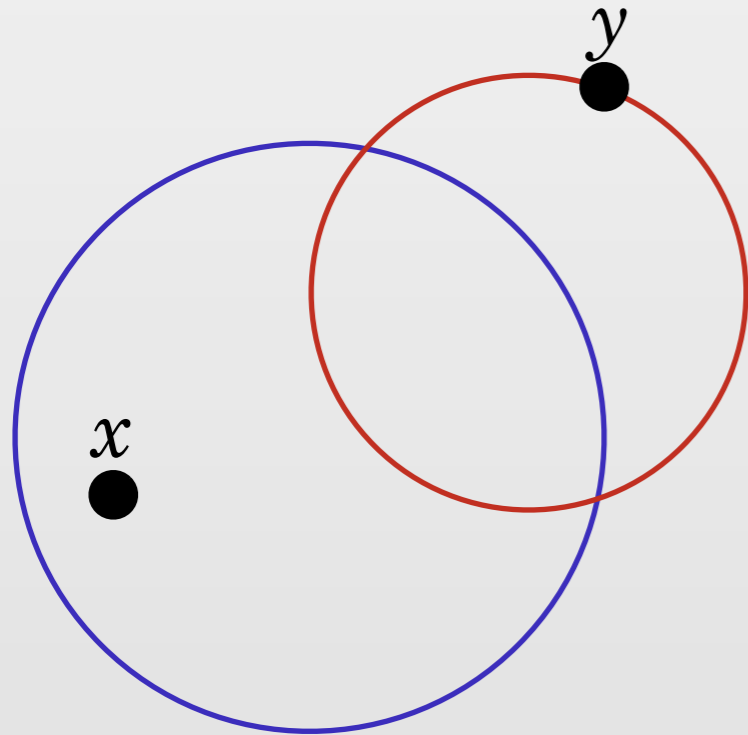
Claim: x only gets touched $O(1)$ times per round.



Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

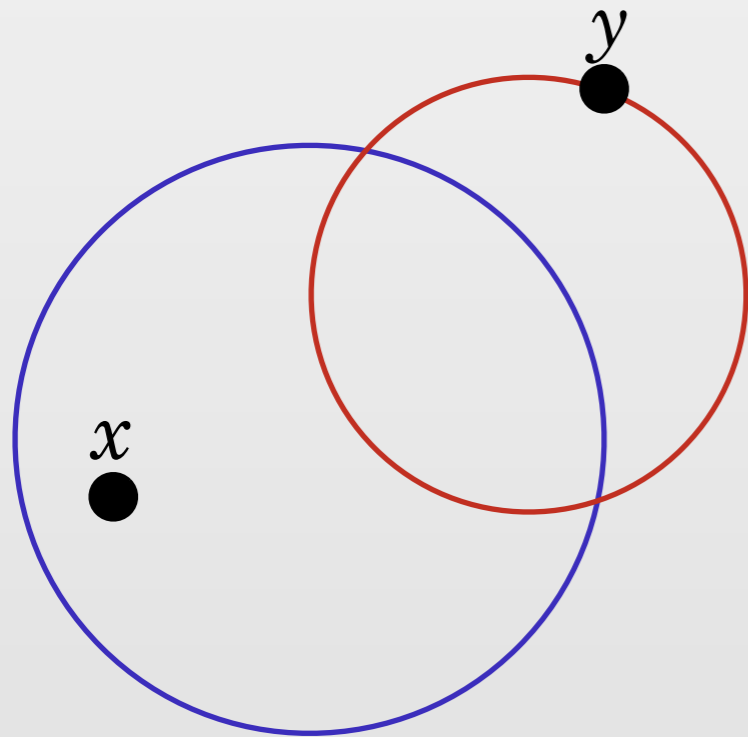
Claim: x only gets touched $O(1)$ times per round.



Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.

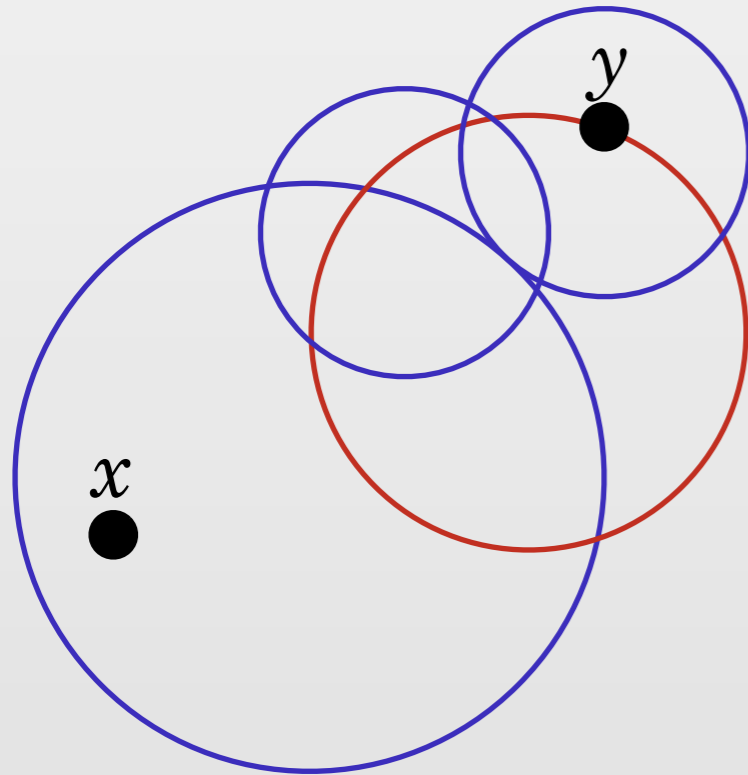


y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.

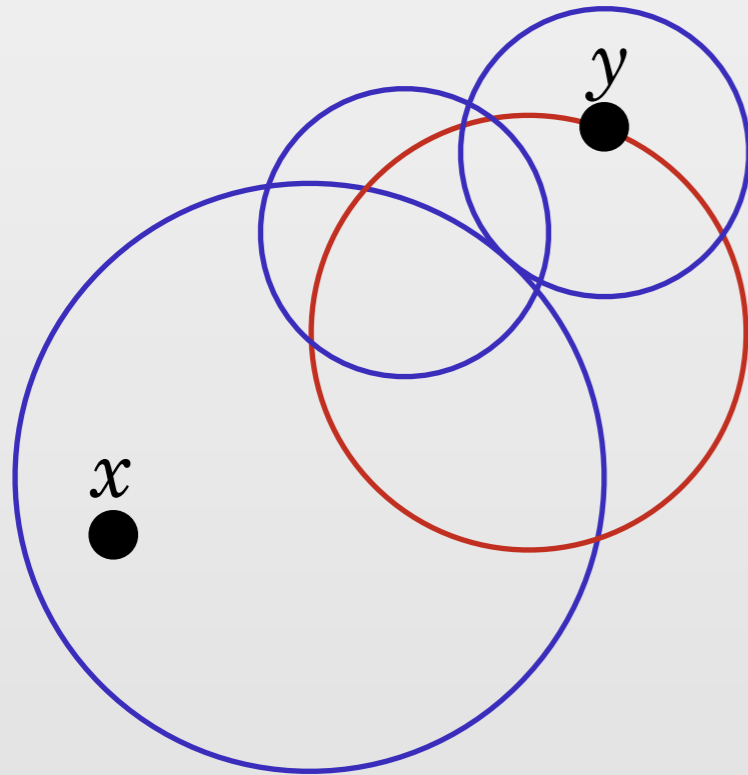


y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.



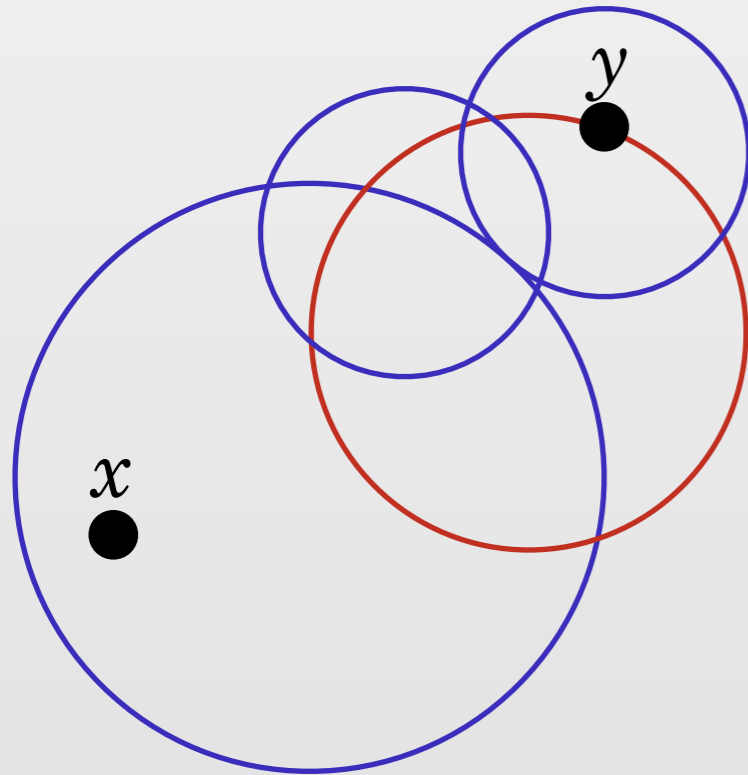
y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

Only $O(1)$ D-balls are within 2 hops.

Amortized Cost of a Round is $O(n)$

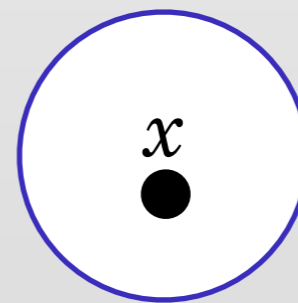
Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.



y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

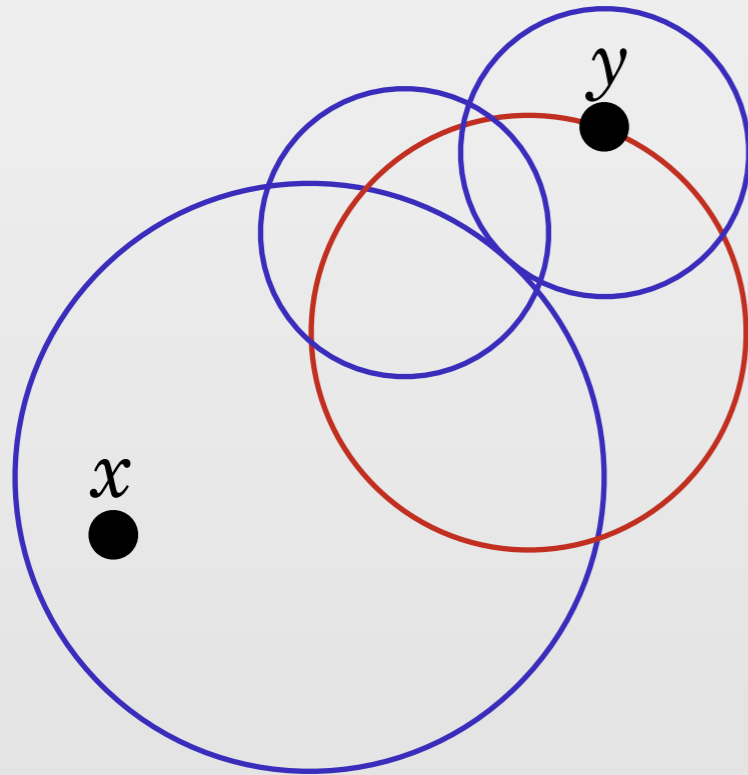
Only $O(1)$ D-balls are within 2 hops.



Amortized Cost of a Round is $O(n)$

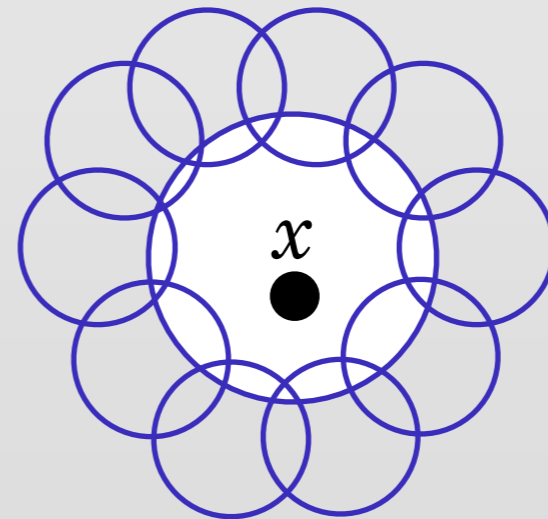
Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.



y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

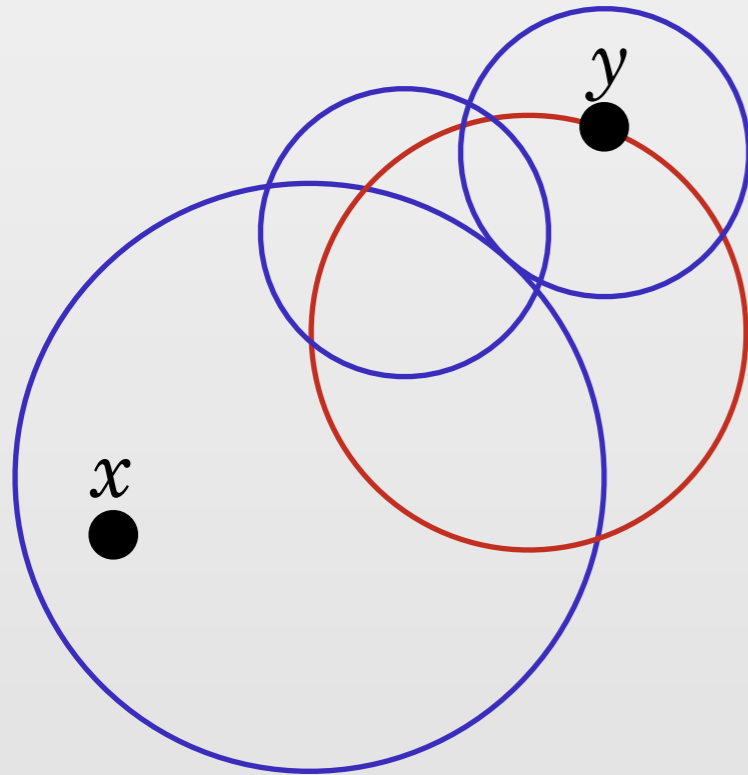
Only $O(1)$ D-balls are within 2 hops.



Amortized Cost of a Round is $O(n)$

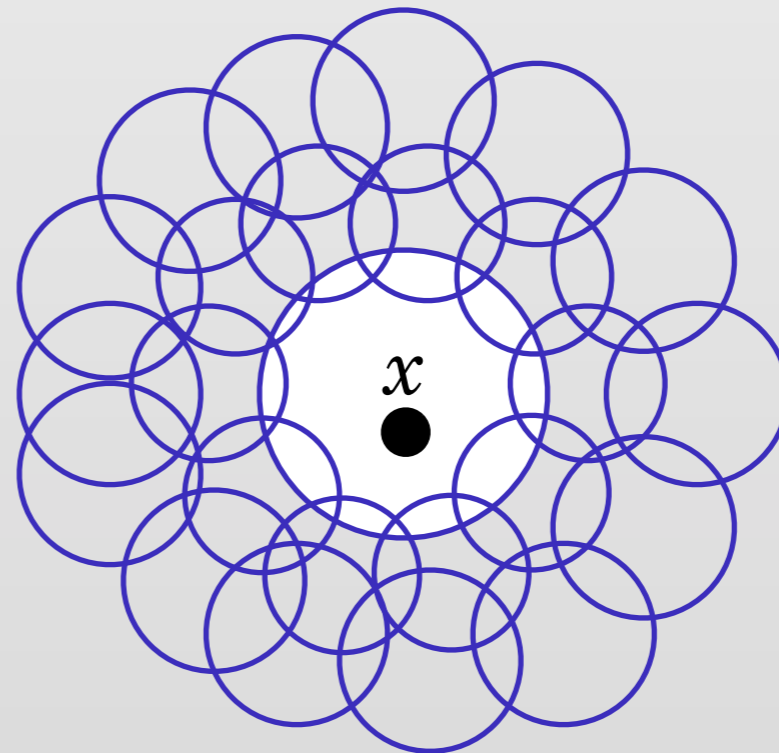
Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.



y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

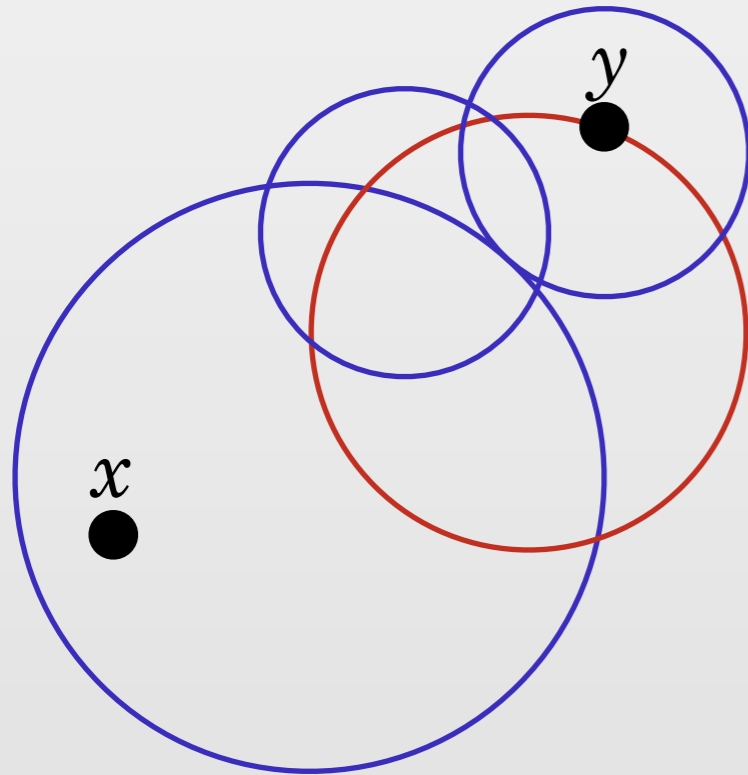
Only $O(1)$ D-balls are within 2 hops.



Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

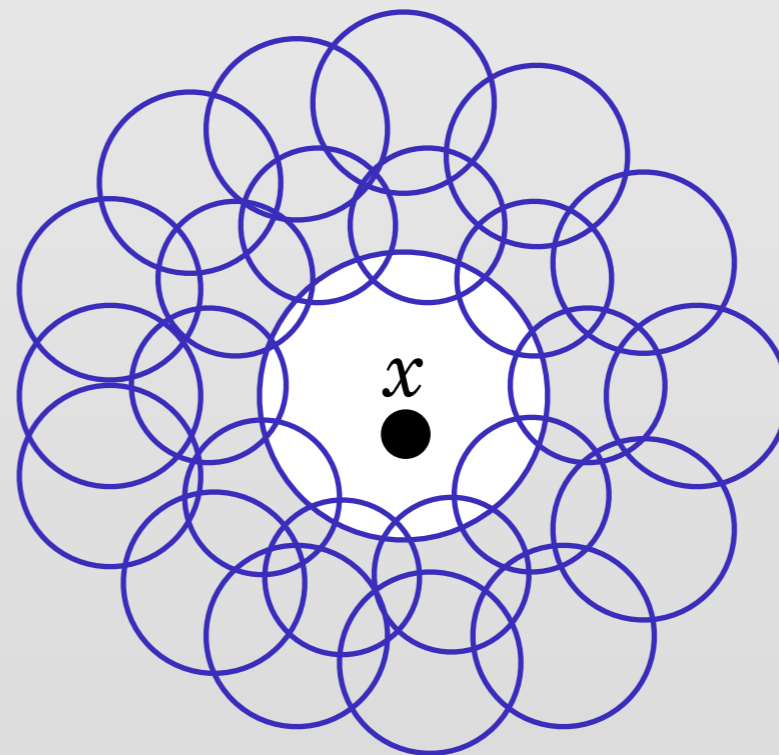
Claim: x only gets touched $O(1)$ times per round.



y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

Only $O(1)$ D-balls are within 2 hops.

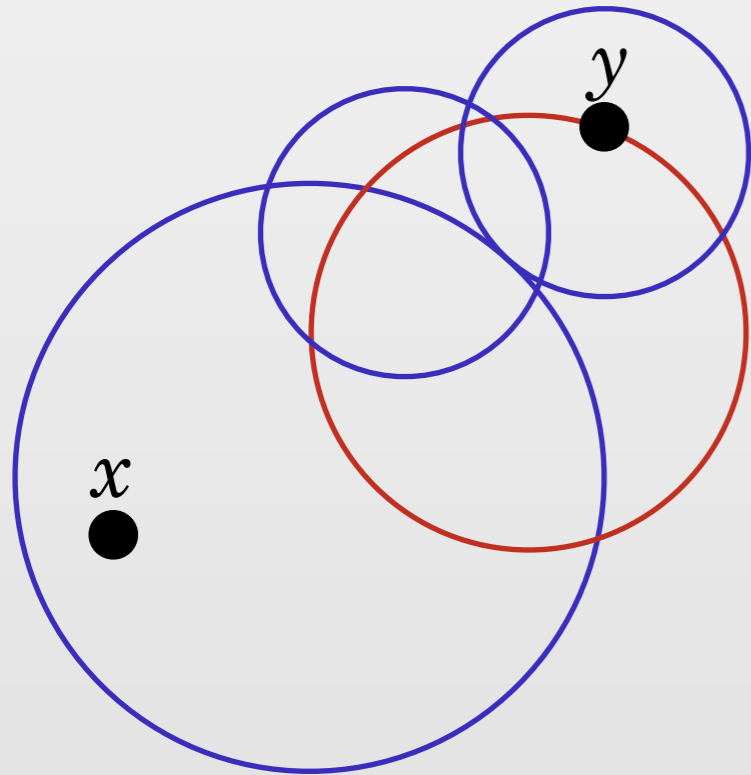
Only $O(1)$ points are added to any D-ball in a round.



Amortized Cost of a Round is $O(n)$

Watch an uninserted point x .

Claim: x only gets touched $O(1)$ times per round.

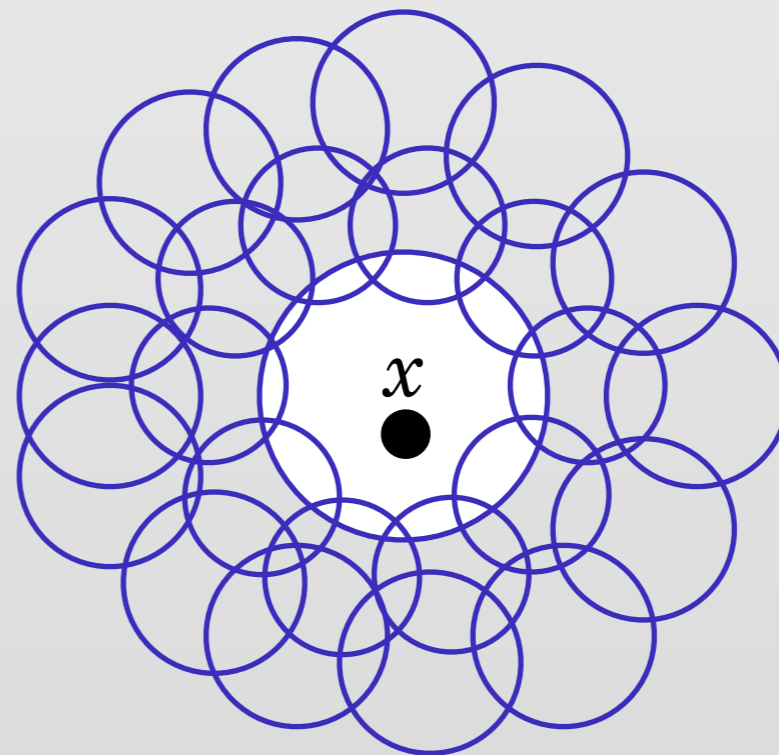


y touches $x \Rightarrow y$ is “close” to x
close = 2 hops among D-balls

Only $O(1)$ D-balls are within 2 hops.

Only $O(1)$ points are added to any D-ball in a round.

$O(n)$ total work per round.



Meshing

Meshing Metric Nets

Meshing
Metric Nets
Geometric D&C

Meshing
Metric Nets
Geometric D&C
Range Nets

Meshing
Metric Nets
Geometric D&C
Range Nets
Ball Covering

Meshing
Metric Nets
Geometric D&C
Range Nets
Ball Covering
Combinatorial D&C

Meshing
Metric Nets
Geometric D&C
Range Nets
Ball Covering
Combinatorial D&C
Amortized Point Location

Meshing

Metric Nets

Geometric D&C

Range Nets

Ball Covering

Combinatorial D&C

Amortized Point Location

Finally, $O(n \log n + m)$ point meshing

Meshing
Metric Nets
Geometric D&C
Range Nets
Ball Covering
Combinatorial D&C
Amortized Point Location
Finally, $O(n \log n + m)$ point meshing

Thanks.

Thanks again.