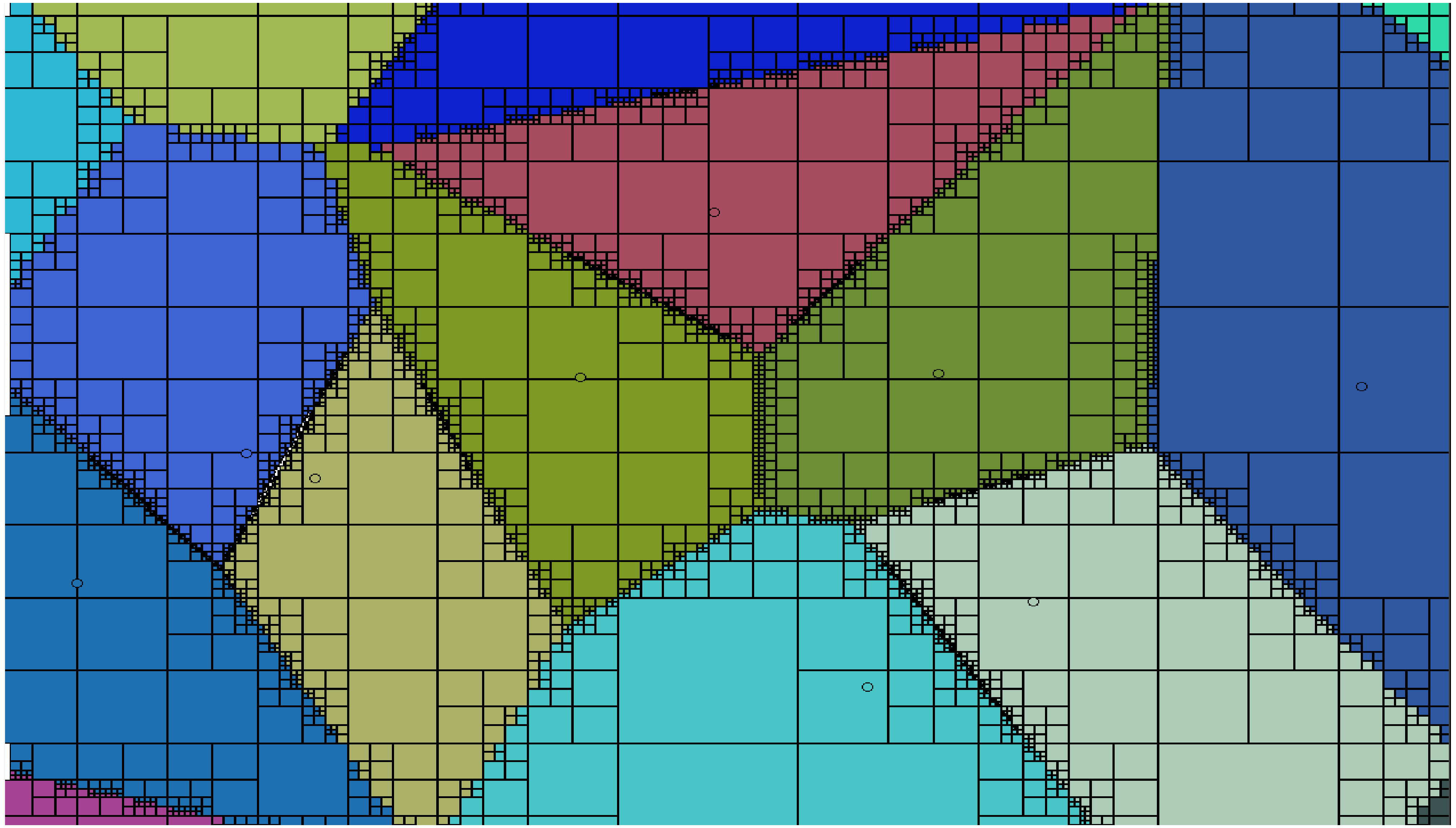
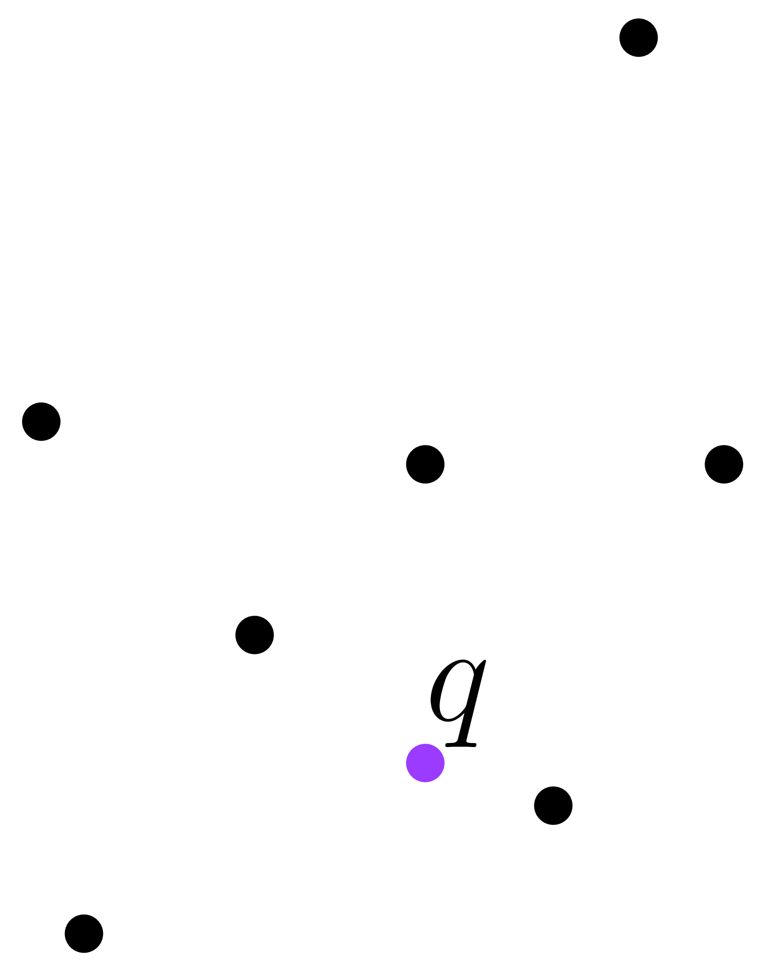
Approximate Voronoi Diagrams



Recap Point Location Among Balls

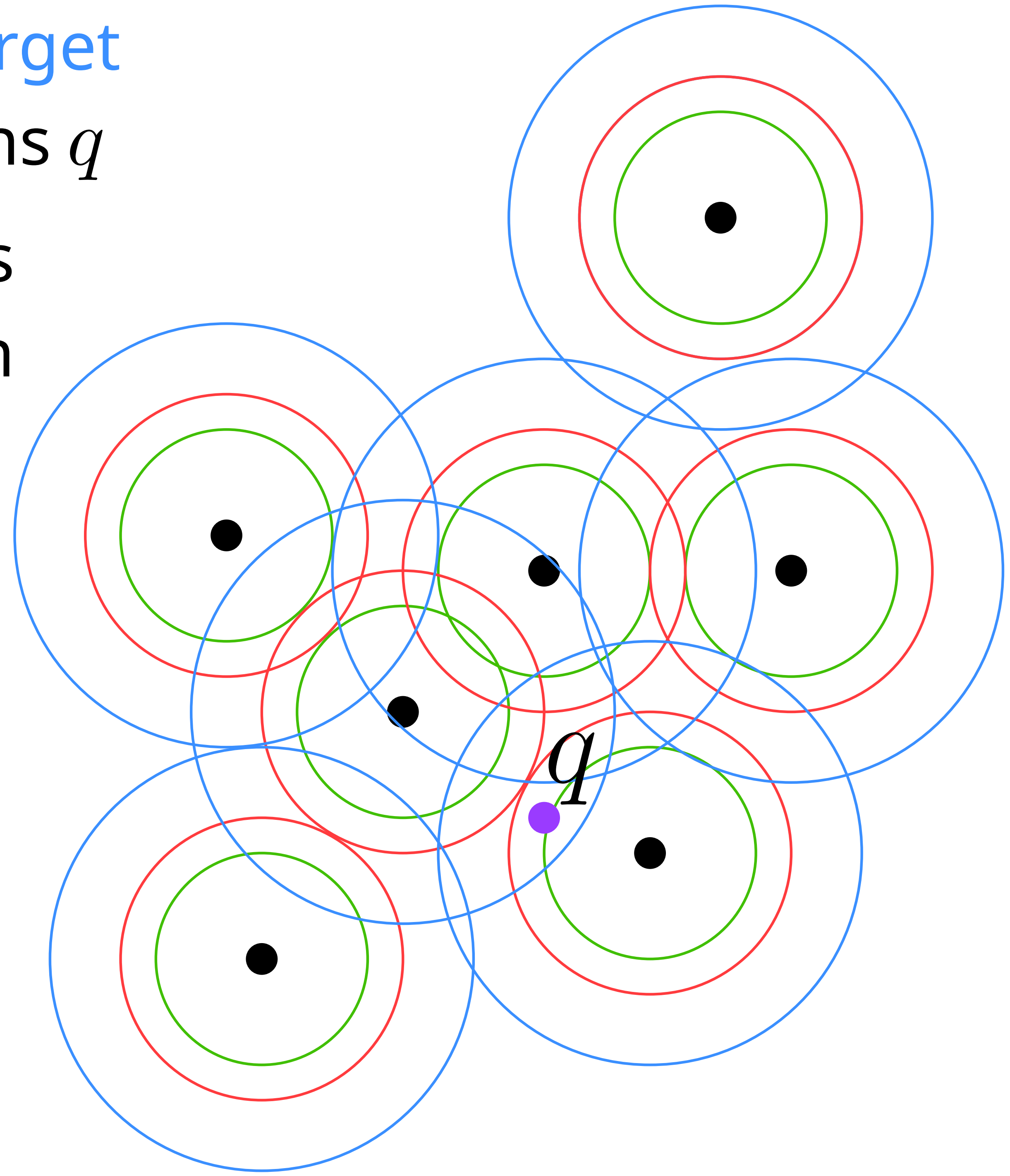
Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q



Recap Point Location Among Balls

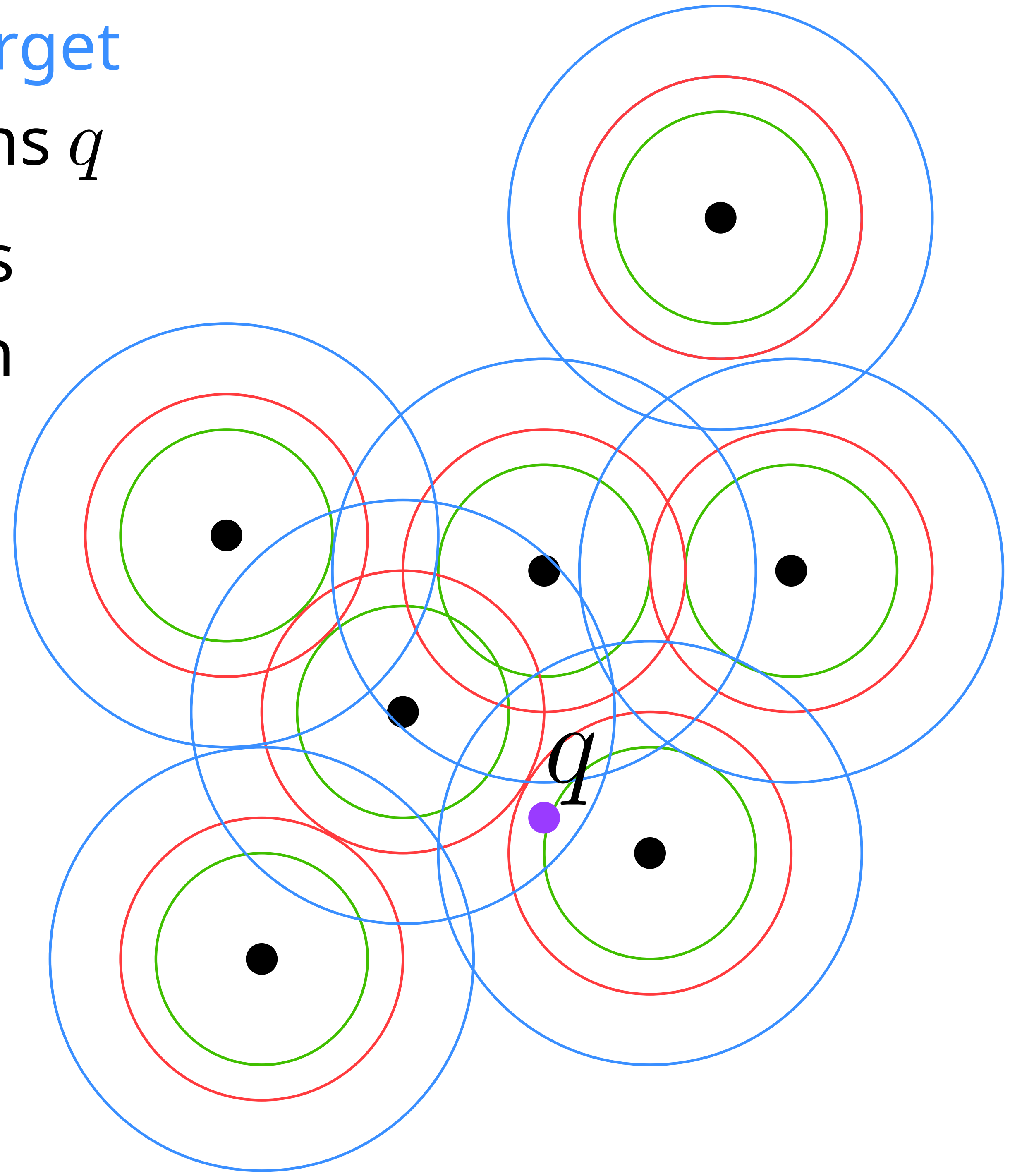
- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R)



Recap Point Location Among Balls

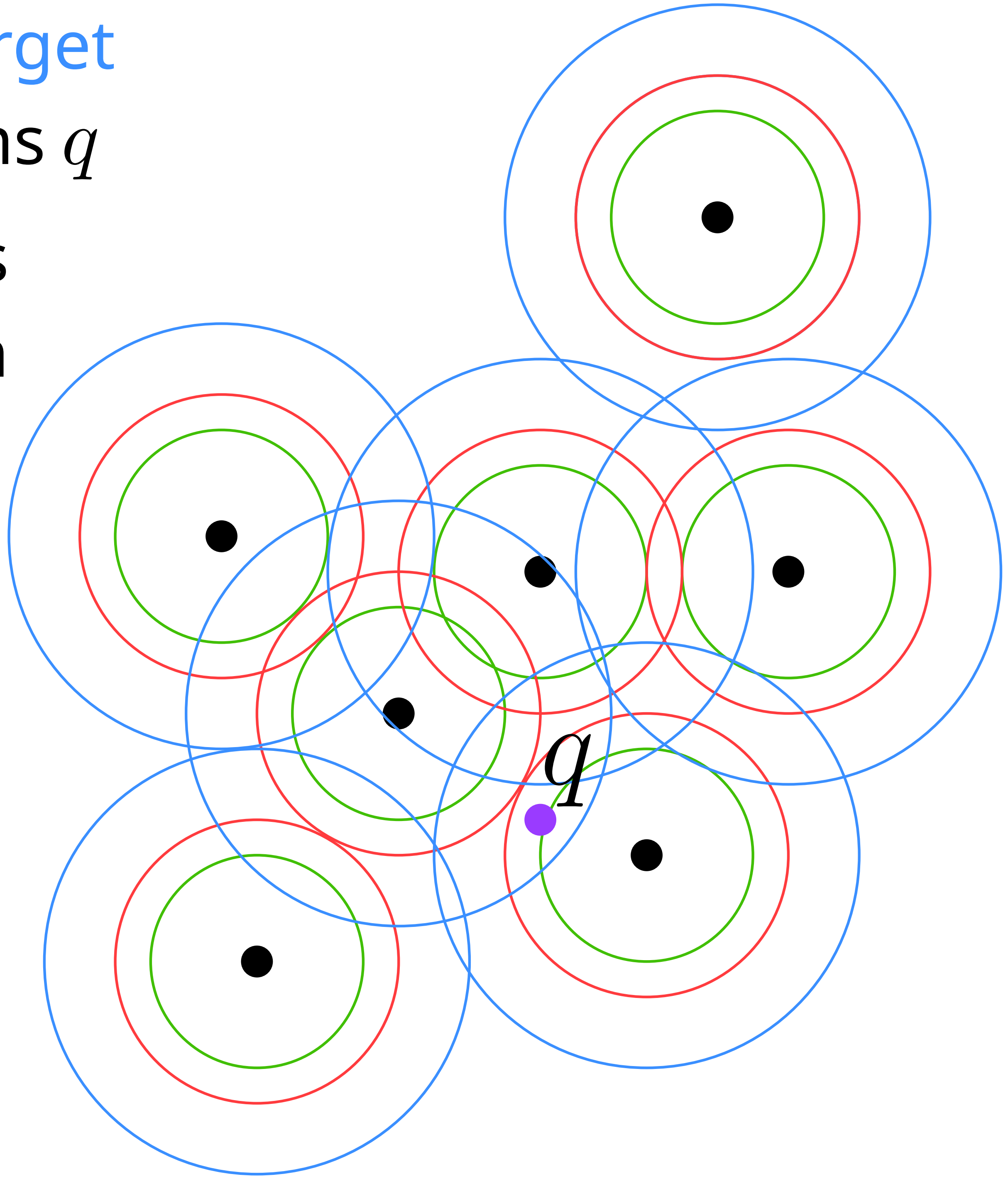
- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R)

Size: ?



Recap Point Location Among Balls

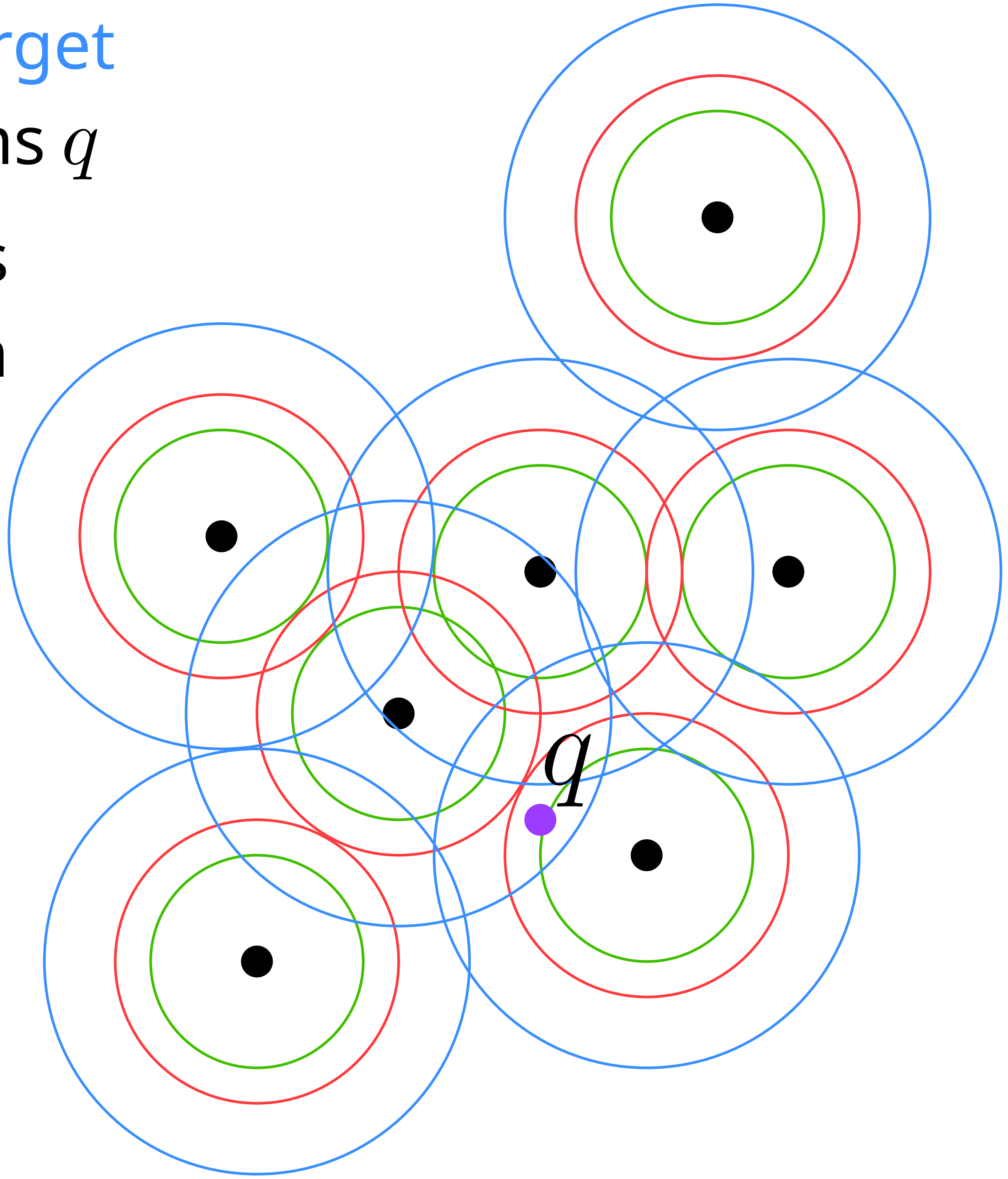
- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$



Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points

How?

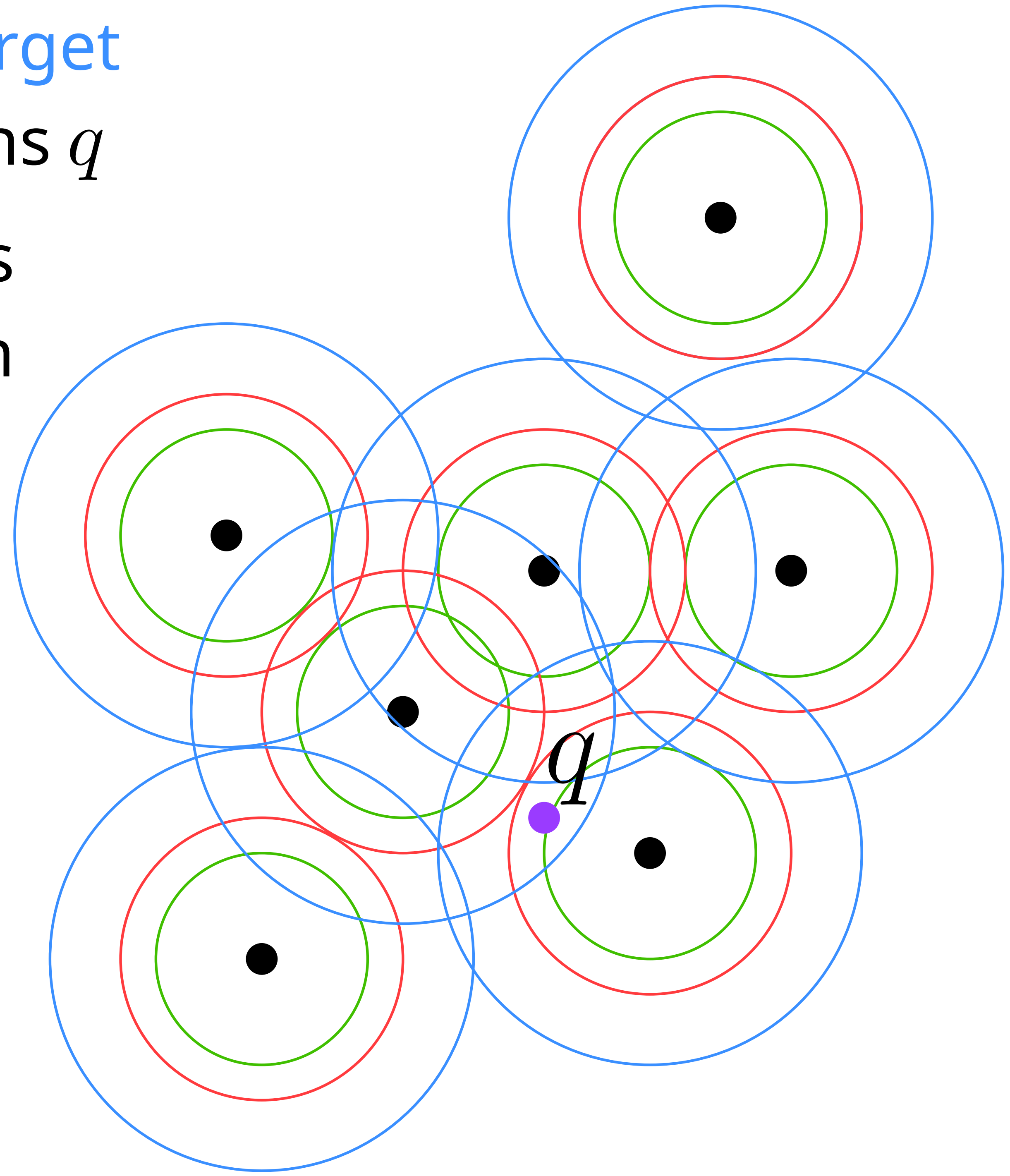


Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points

How?

  - bottom-up: compute MST, lowest to heighest weight: merge components

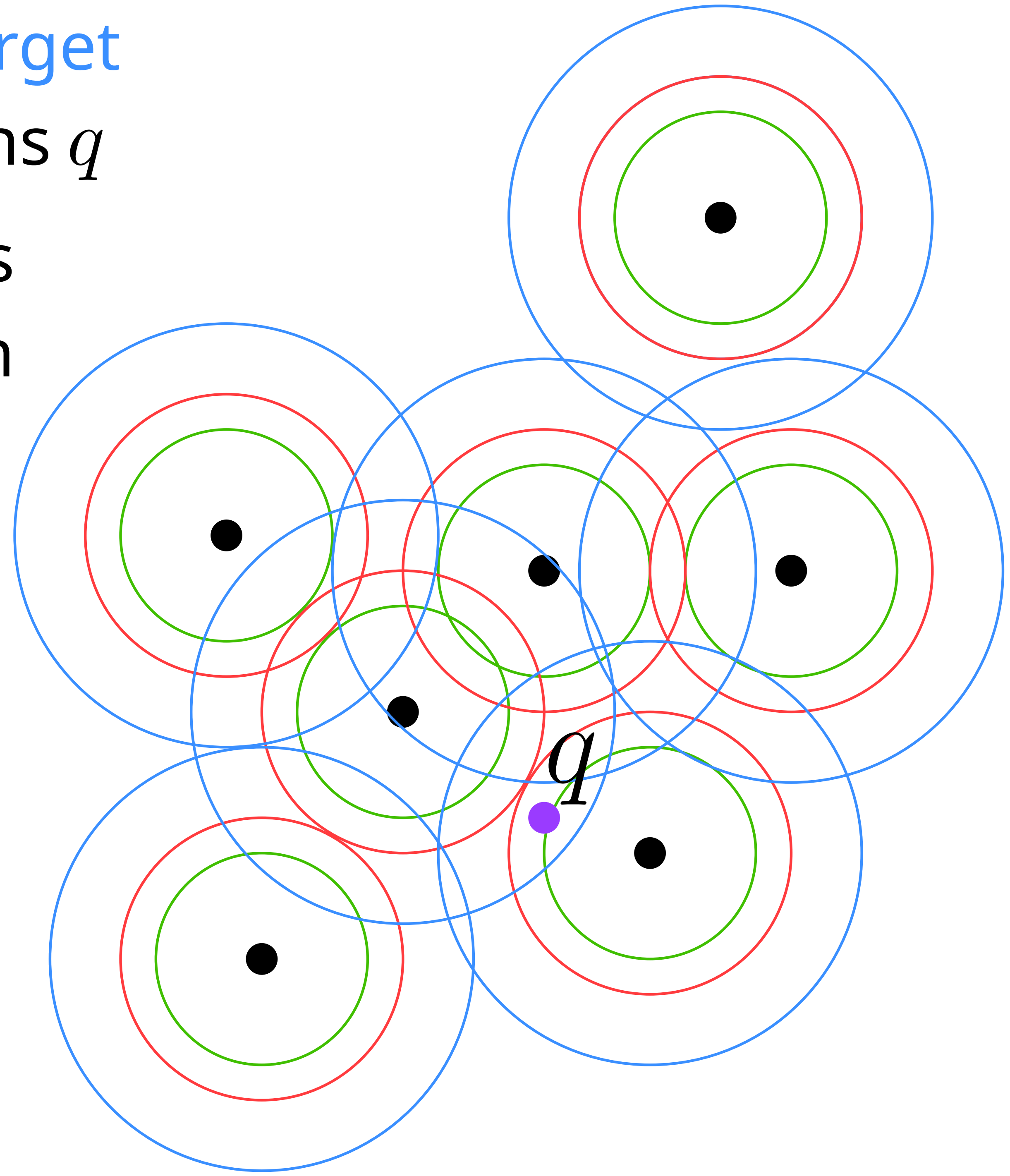


Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points

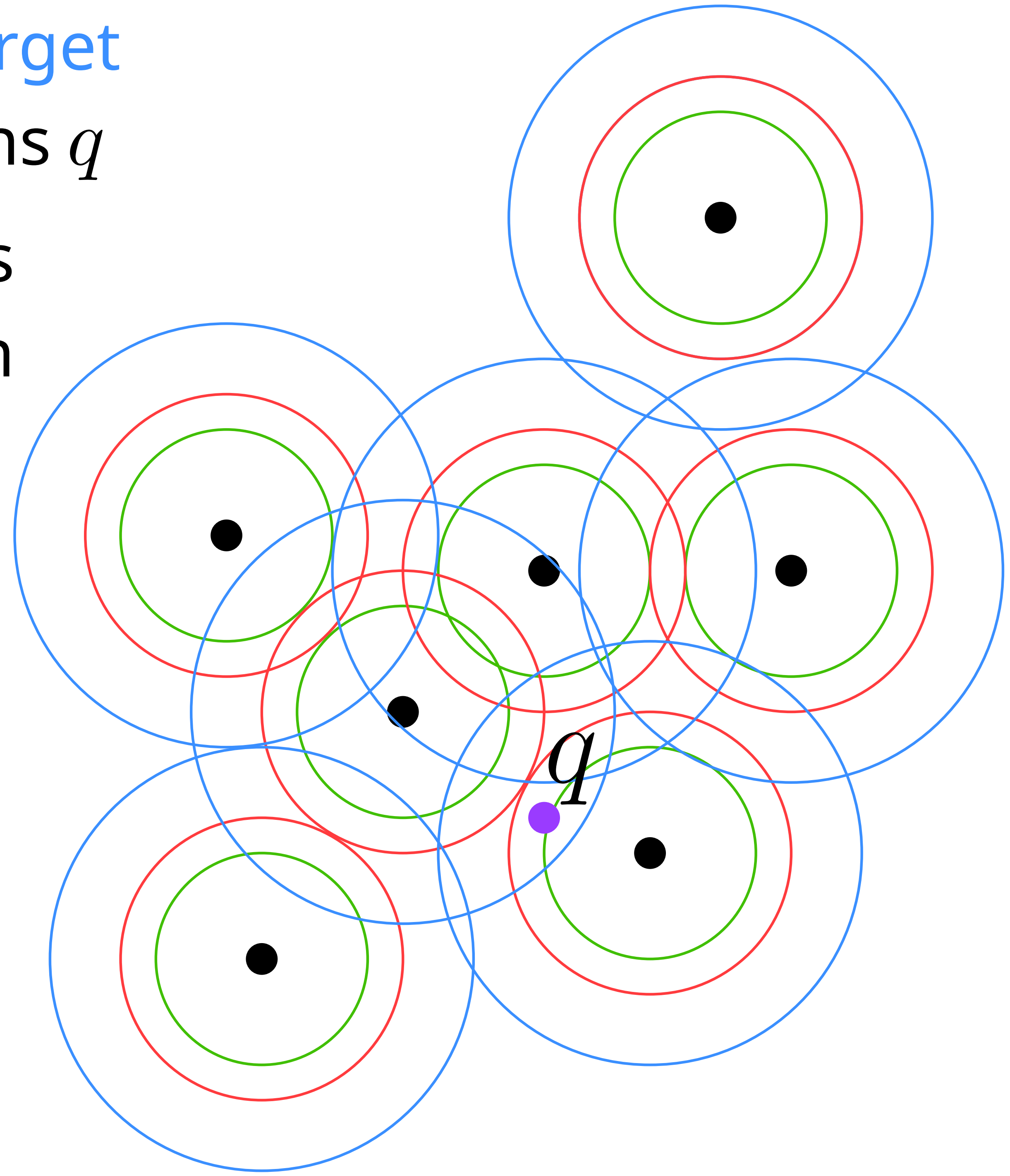
How?

- bottom-up: compute MST, lowest to heighest weight: merge components
- Euclidean space: shifted quadtree



Recap Point Location Among Balls

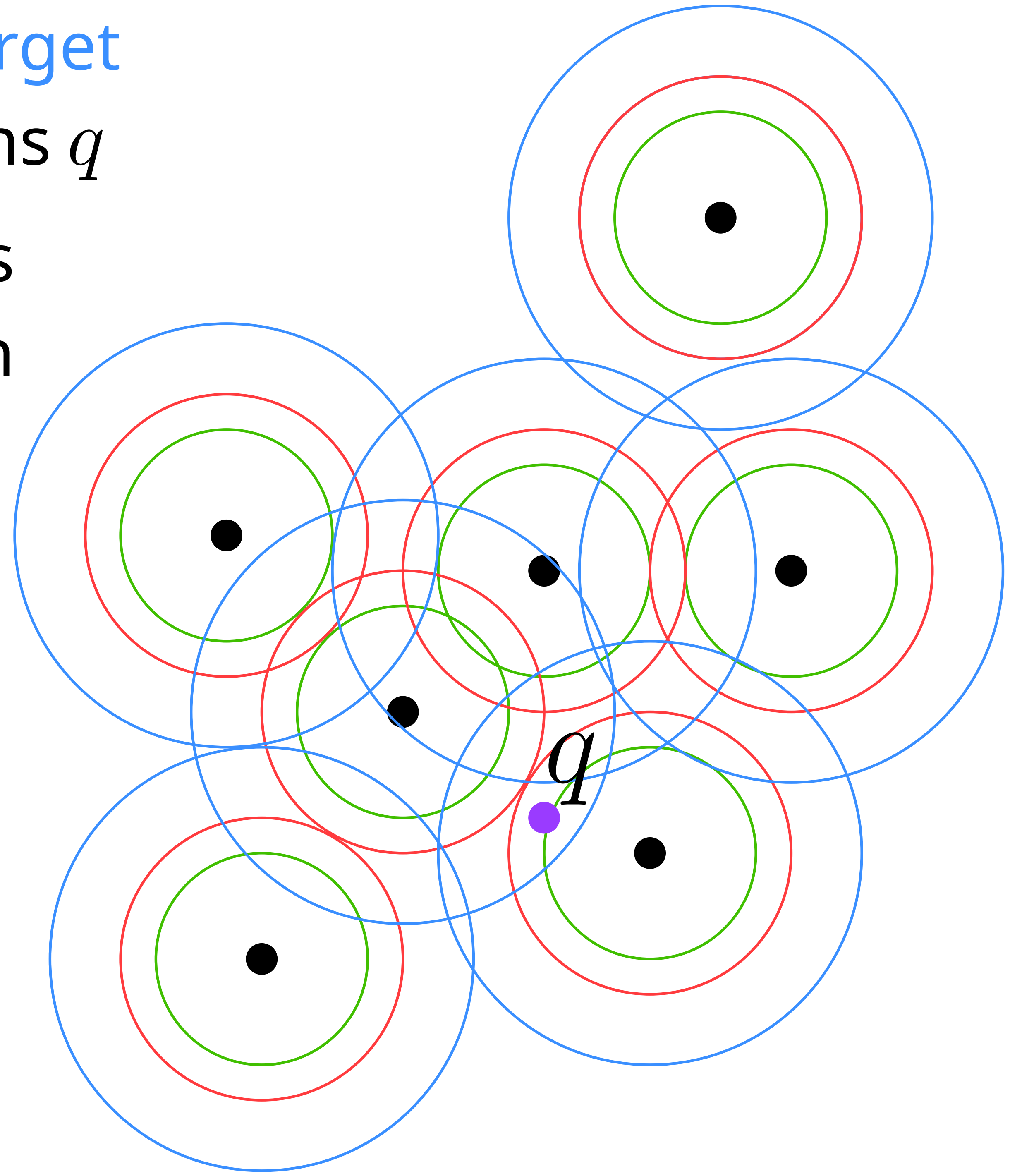
- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon) \log n \log(n/\varepsilon))$



Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon)\log n \log(n/\varepsilon))$

per interval structure:
$O(n_v/\varepsilon \log(n^{O(1)}/\varepsilon) = O(n_v/\varepsilon \log(n/\varepsilon))$

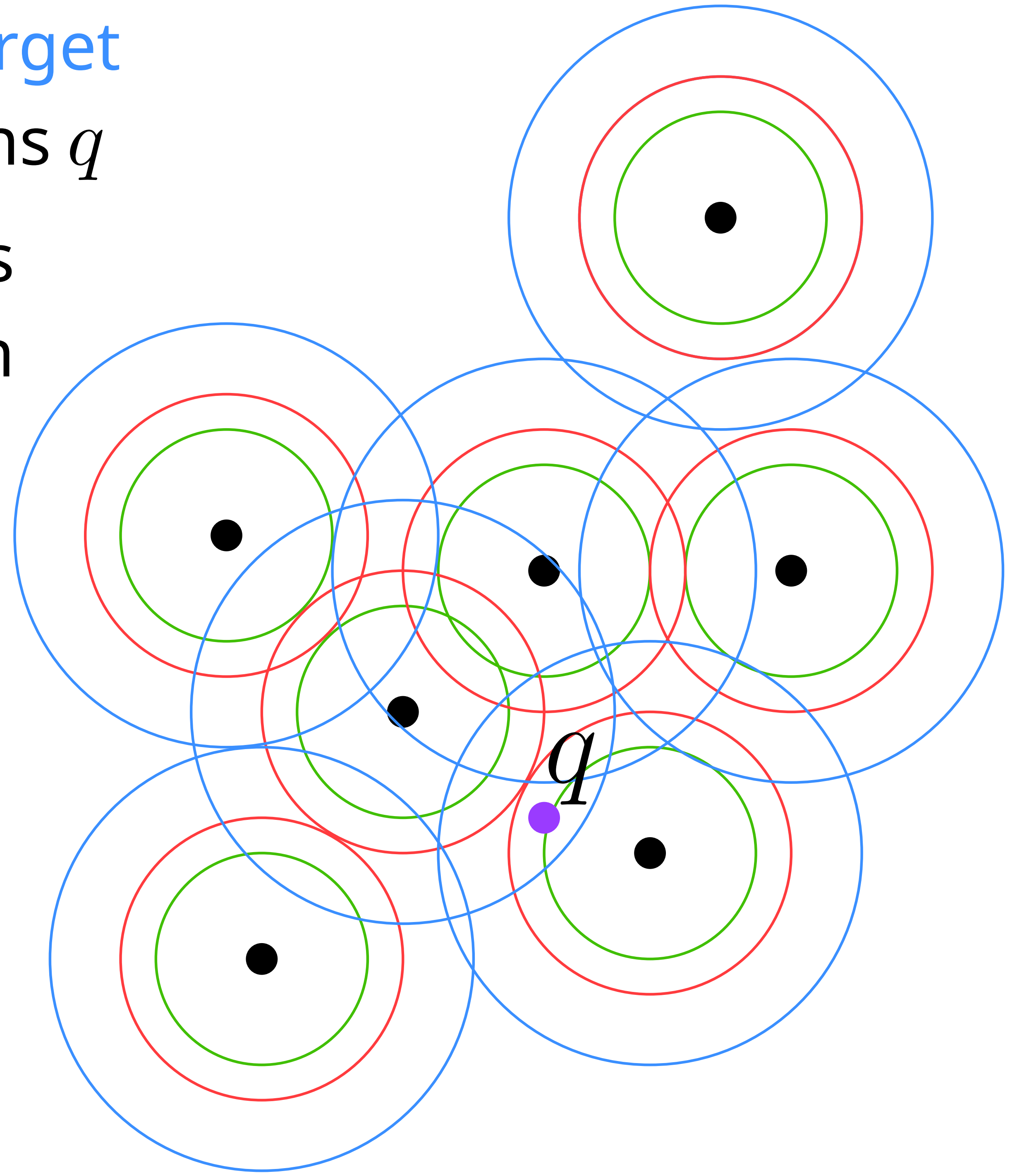


Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon) \log n \log(n/\varepsilon))$

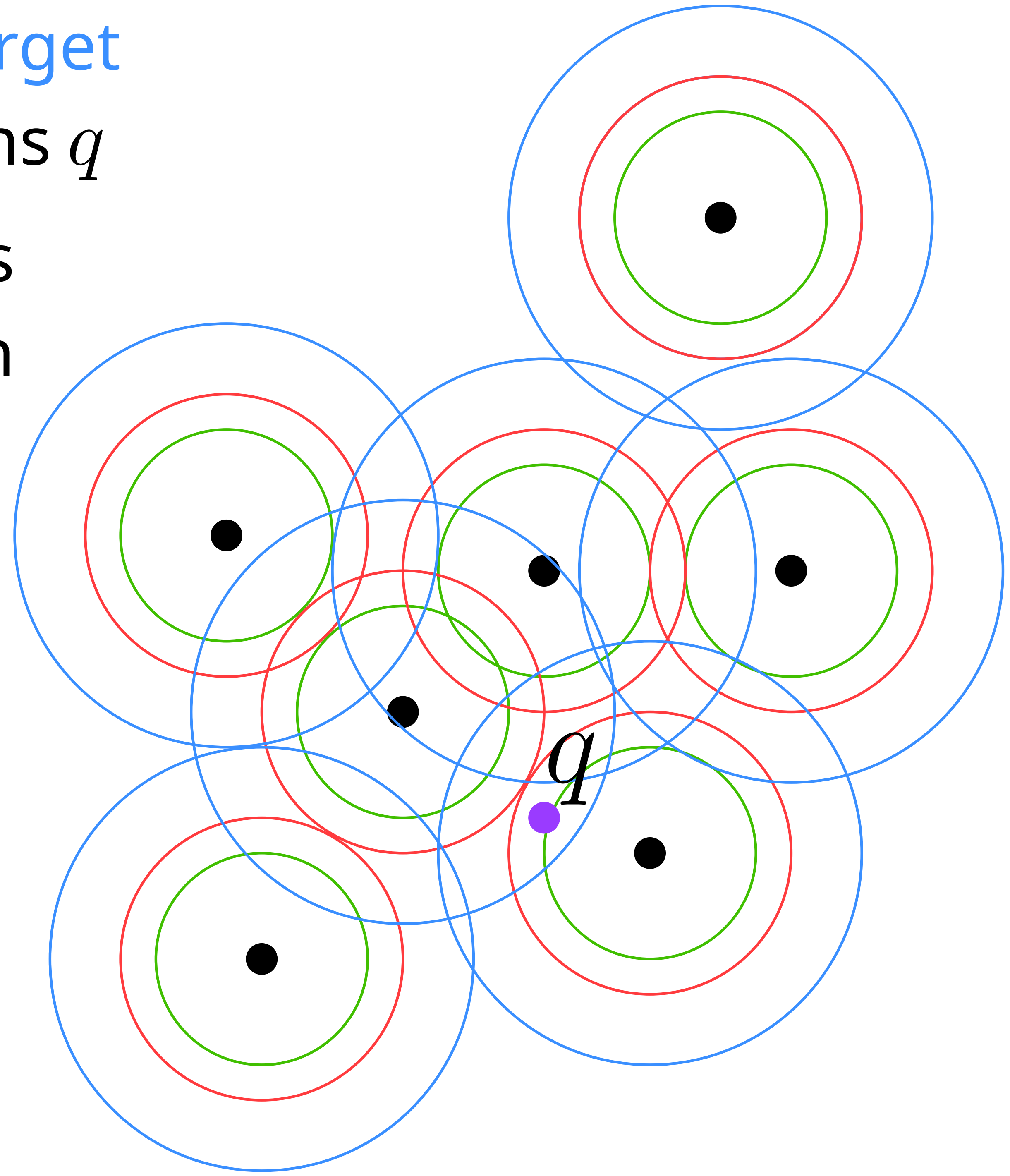
per interval structure:
$O(n_v/\varepsilon \log(n^{O(1)}/\varepsilon) = O(n_v/\varepsilon \log(n/\varepsilon))$

points occur up to $\log n$ times



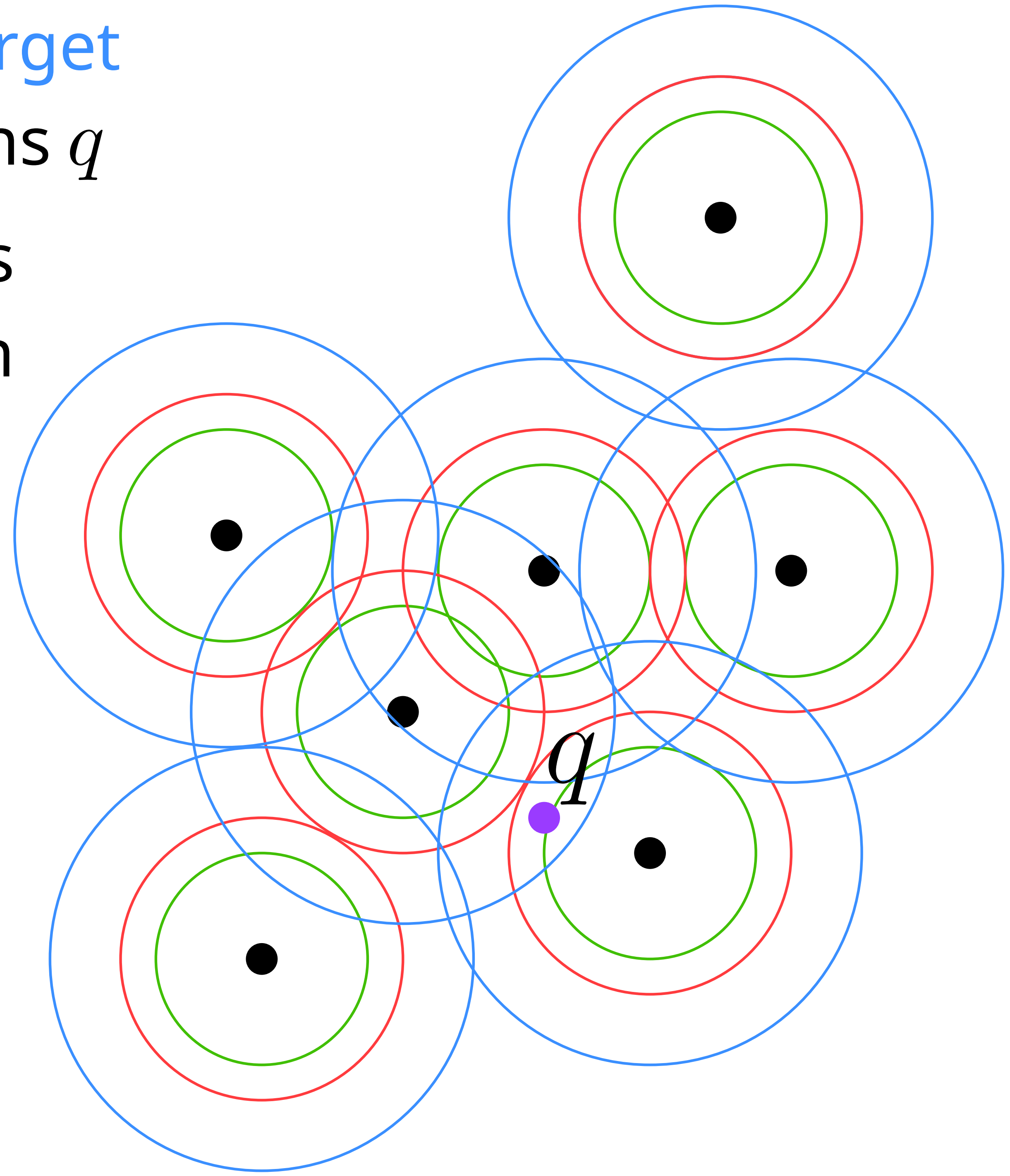
Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon)\log n \log(n/\varepsilon))$
  improved (book): $O((n/\varepsilon)\log n)$



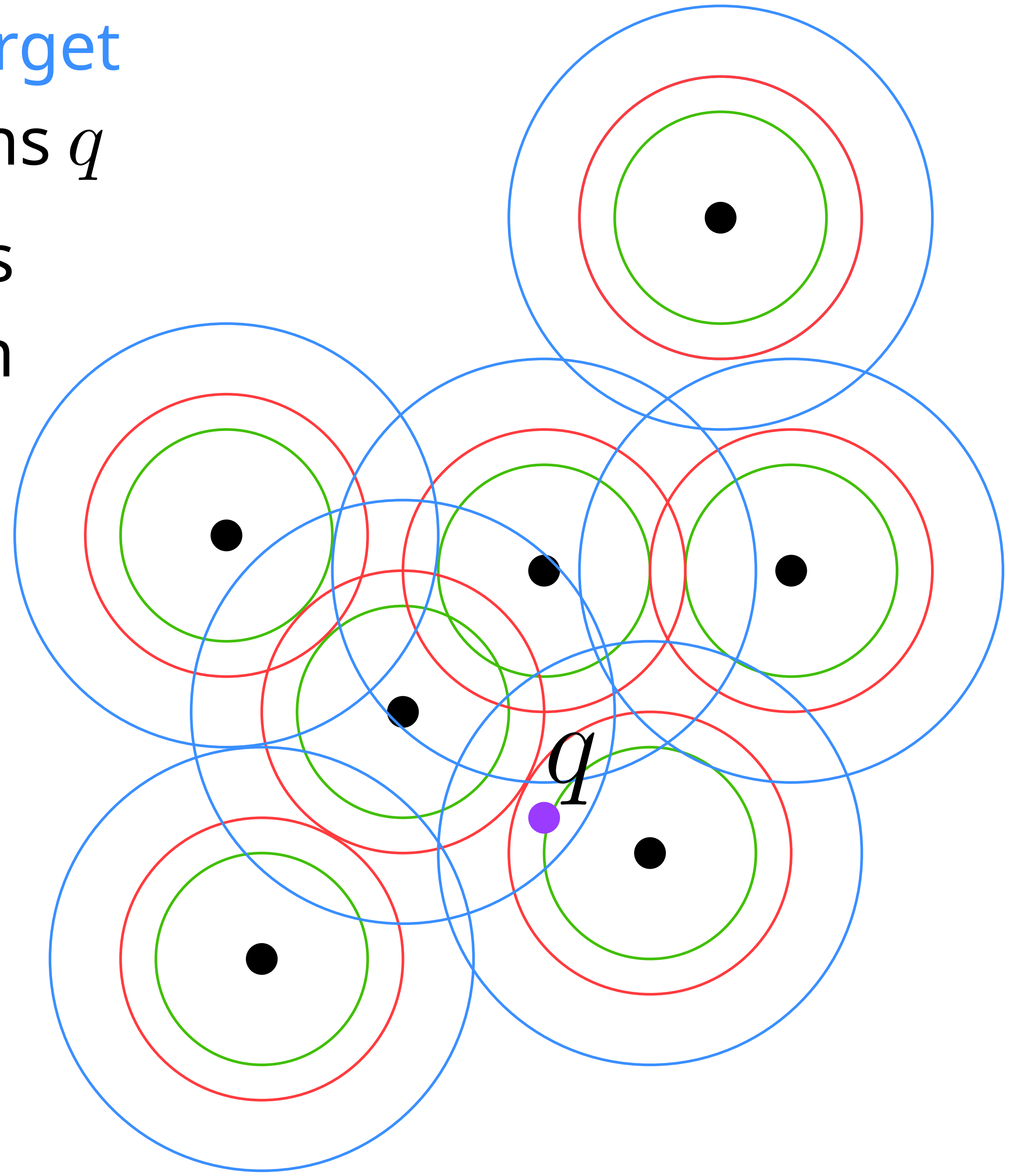
Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon)\log n \log(n/\varepsilon))$
  improved (book): $O((n/\varepsilon)\log n)$
- **# of near-neighbor queries:** $O(\log(n/\varepsilon))$



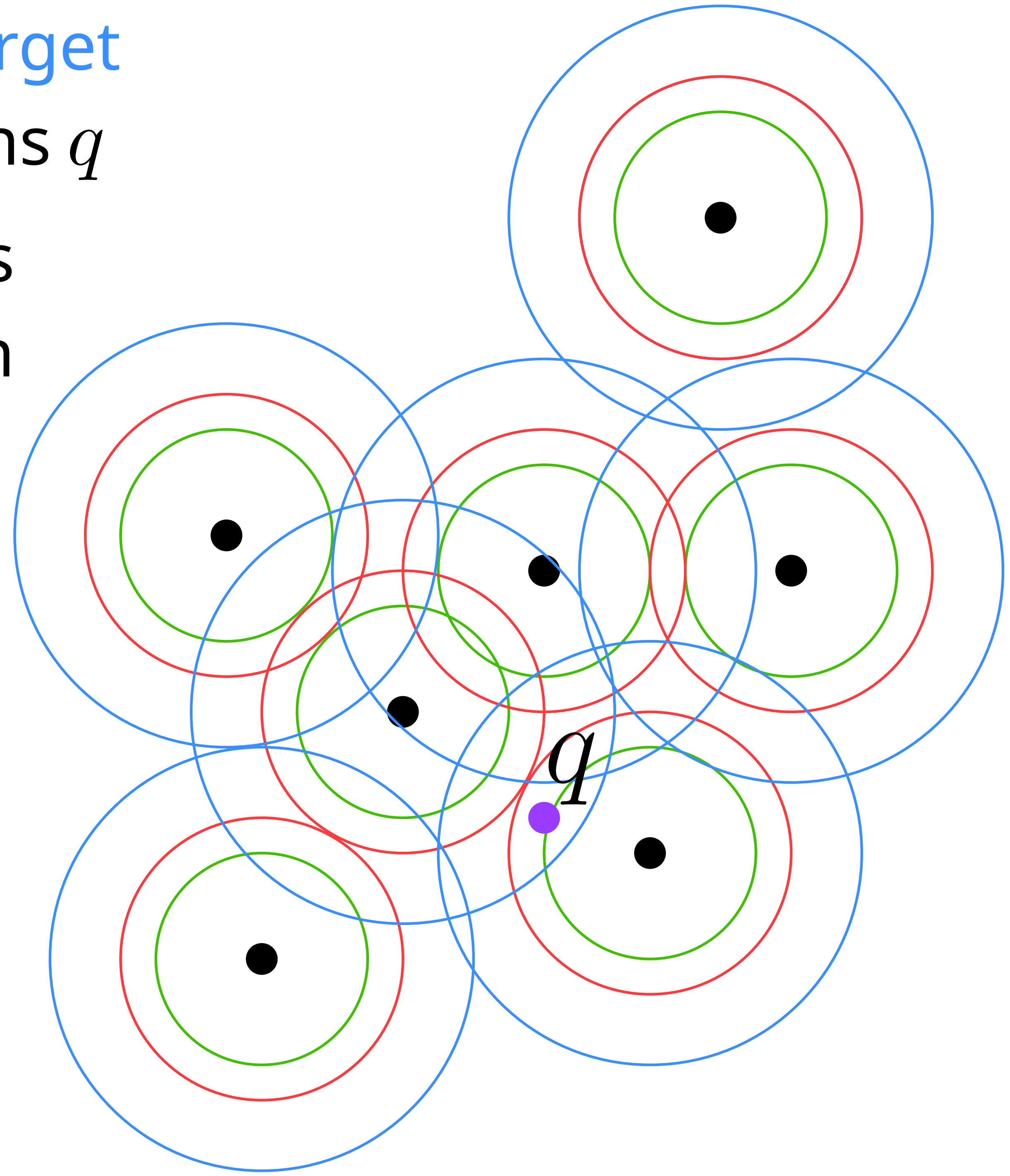
Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon)\log n \log(n/\varepsilon))$
  improved (book): $O((n/\varepsilon)\log n)$
- **# of near-neighbor queries:** $O(\log(n/\varepsilon))$
  $\log n$ times only against r_v and R_v



Recap Point Location Among Balls

- Given a point set P and a query point q, the target ball $\odot_{\mathcal{B}}$ of q is the smallest ball of $\mathcal{B}$ that contains q
- Interval structure $I(P, r, R, \varepsilon)$: Create rings around each point of increasing radii $(1+\varepsilon)^i$ in interval (r, R) Size: $O\left(n/\varepsilon \log(R/r)\right)$
- Create a Balanced Hierarchically Separated Tree (BHST) from the points
- **Space complexity:** $O((n/\varepsilon)\log n \log(n/\varepsilon))$
  improved (book): $O((n/\varepsilon)\log n)$
- **# of near-neighbor queries:** $O(\log{(n/\varepsilon)})$
  $\log n$ times only against r_v and R_v
  once $[r_v, R_v)$: $O(\log(n/\varepsilon))$



Caveat

$O(\log(n/\varepsilon))$ queries

Caveat

$O(\log{(n/\varepsilon)})$ queries

Those queries are also hard . . .

Approximate balls

- For a ball $b = b(p, r)$ the ball $b_{\approx}$ is a $(1+\varepsilon)$-approximation to b if $b \subseteq b_{\approx} \subset b(p, (1+\varepsilon)r)$

Approximate balls

- For a ball $b = b(p, r)$ the ball $b_{\approx}$ is a $(1 + \varepsilon)$-approximation to b if $b \subseteq b_{\approx} \subset b(p, (1 + \varepsilon)r)$
- For a set of balls $\mathcal{B}$, $\mathcal{B}_{\approx}$ is a $(1 + \varepsilon)$-approximation if for all $b \in \mathcal{B}$ there is an approximation $b_{\approx} \in \mathcal{B}_{\approx}$

Approximate balls

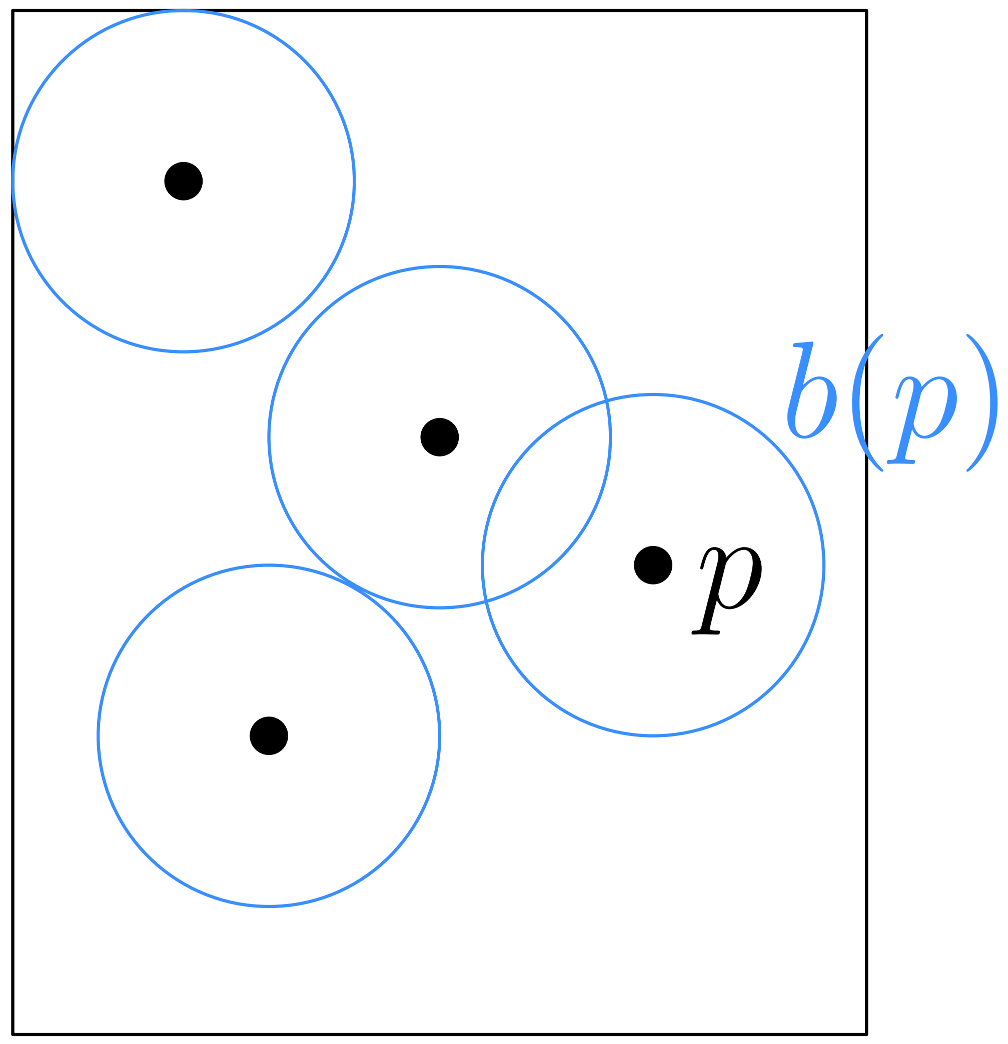
- For a ball $b = b(p, r)$ the ball $b_\approx$ is a $(1+\varepsilon)$-approximation to b if $b \subseteq b_\approx \subset b(p, (1+\varepsilon)r)$
- For a set of balls $\mathcal{B}$, $\mathcal{B}_\approx$ is a $(1+\varepsilon)$-approximation if for all $b \in \mathcal{B}$ there is an approximation $b_\approx \in \mathcal{B}_\approx$
- How should we approximate?

Approximate balls

- For a ball $b = b(p, r)$ the ball $b_{\approx}$ is a $(1+\varepsilon)$-approximation to b if $b \subseteq b_{\approx} \subset b(p, (1+\varepsilon)r)$
- For a set of balls $\mathcal{B}$, $\mathcal{B}_{\approx}$ is a $(1+\varepsilon)$-approximation if for all $b \in \mathcal{B}$ there is an approximation $b_{\approx} \in \mathcal{B}_{\approx}$
- How should we approximate?

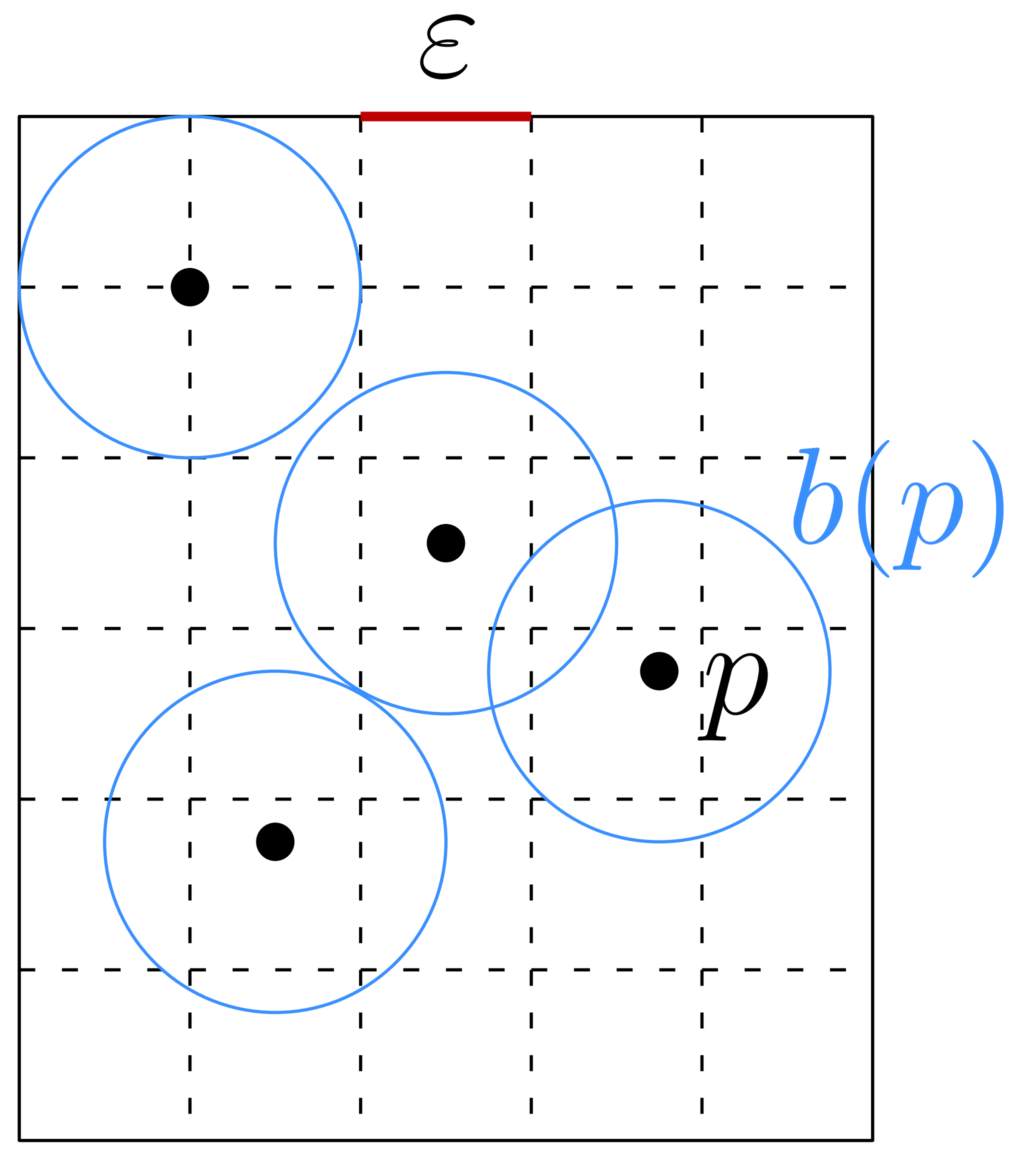
The power of grids!

Approximating the ball



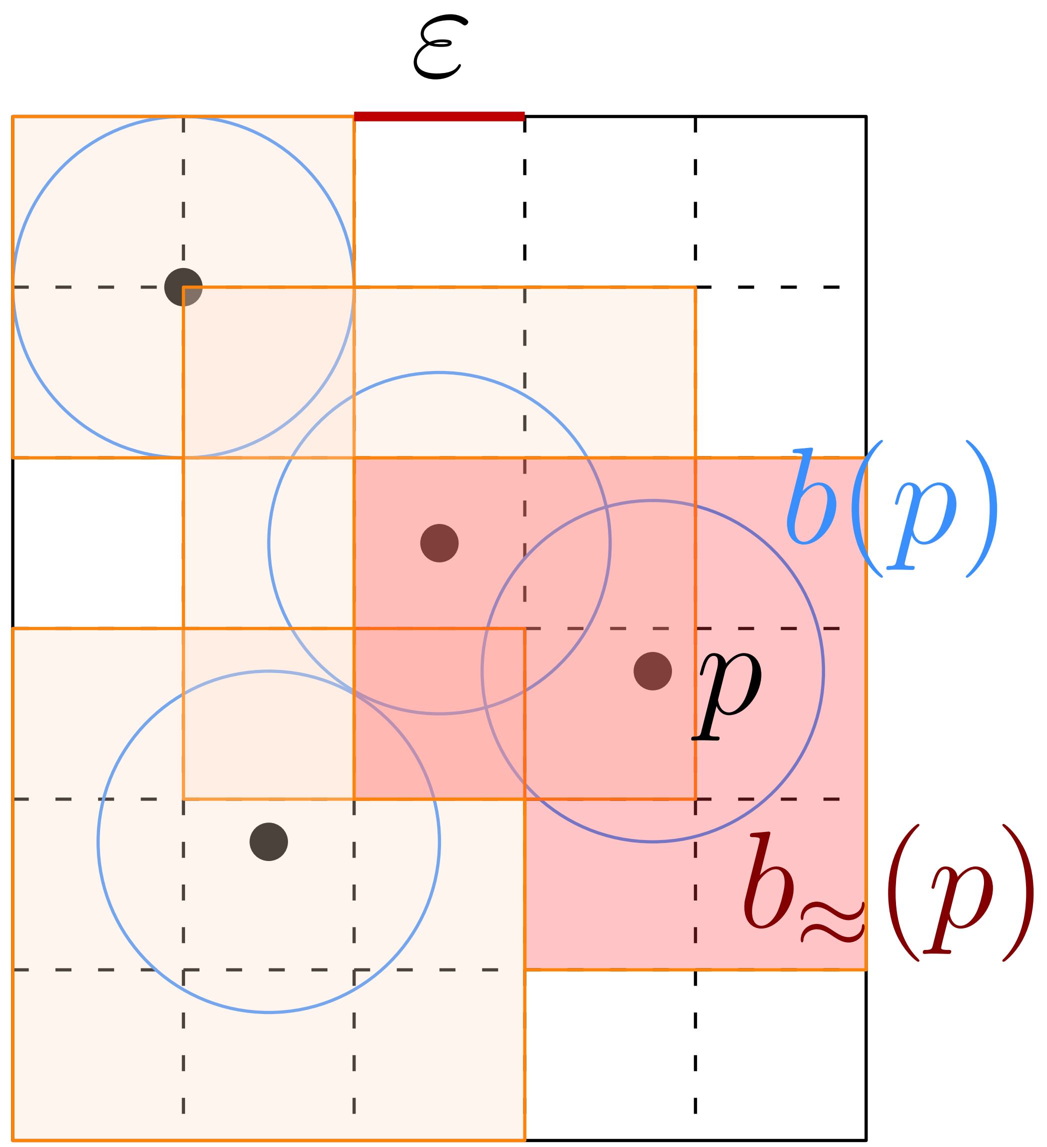
Approximating the ball

- Divide the space into a grid with sides ε



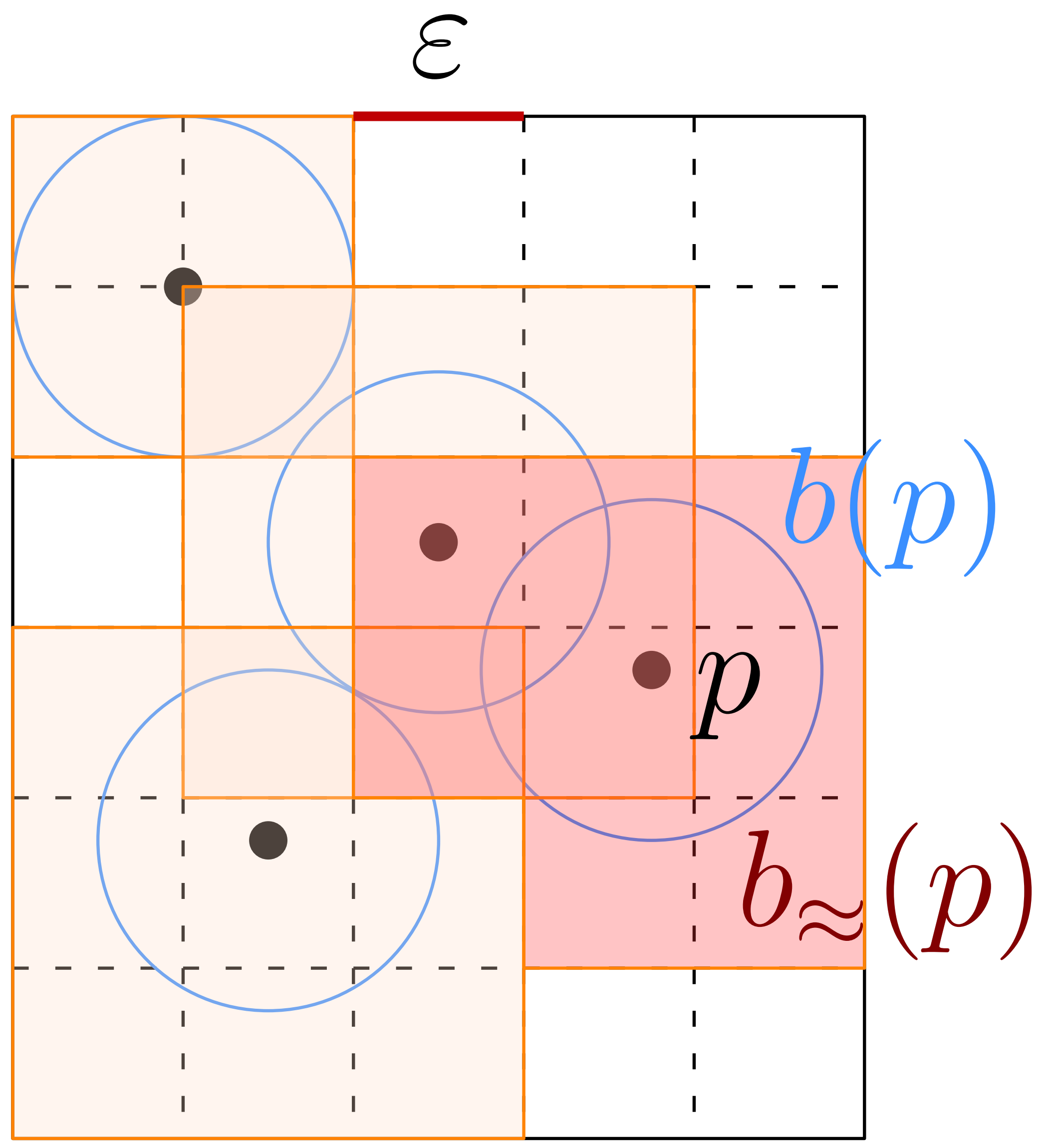
Approximating the ball

- Divide the space into a grid with sides ε
- Define $b_{\approx}(p)$ as the grid cells intersected by $b(p)$



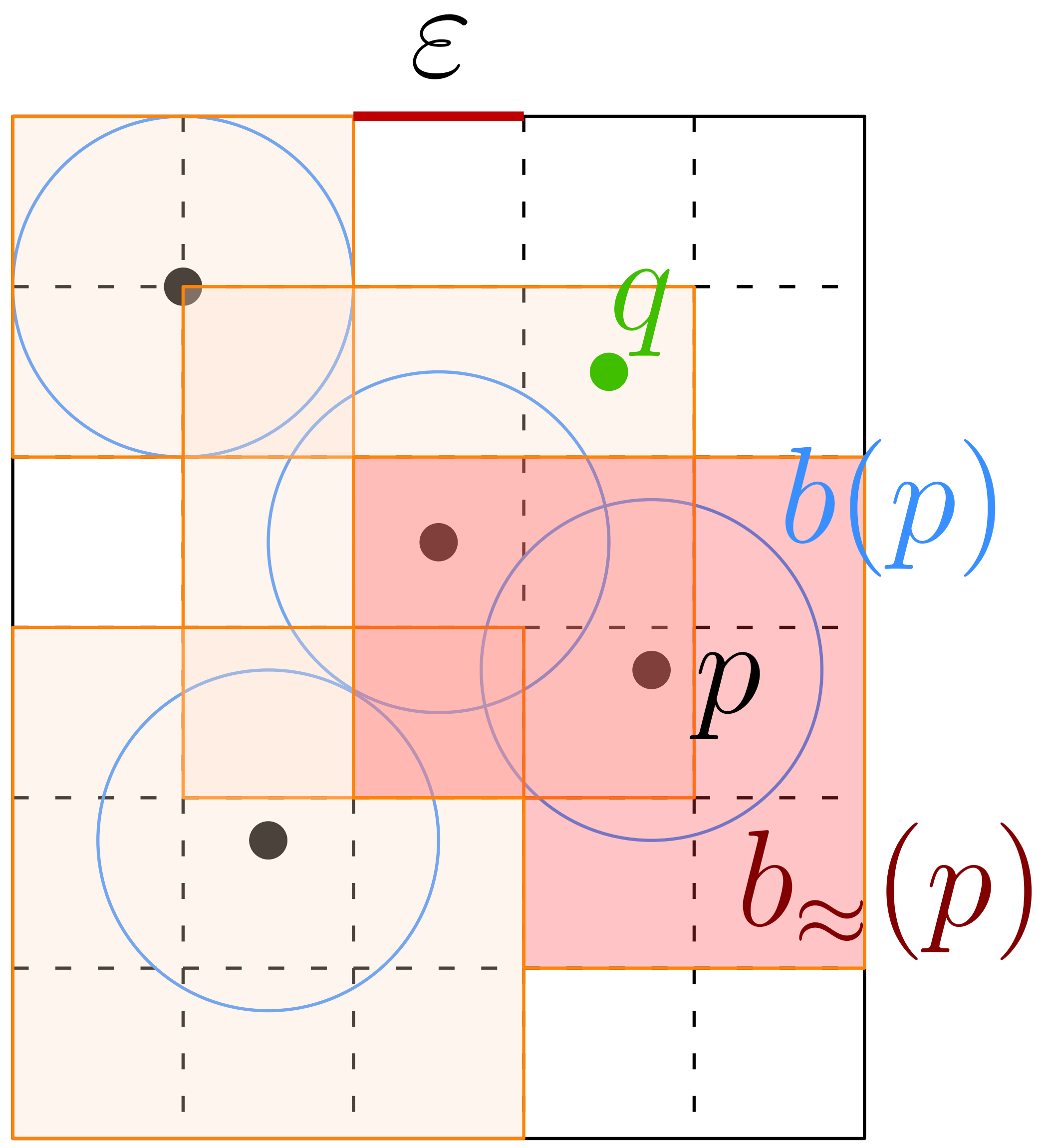
Approximating the ball

- Divide the space into a grid with sides ε
- Define $b_{\approx}(p)$ as the grid cells intersected by $b(p)$
- Throw all $b_{\approx}$ into a hashtable



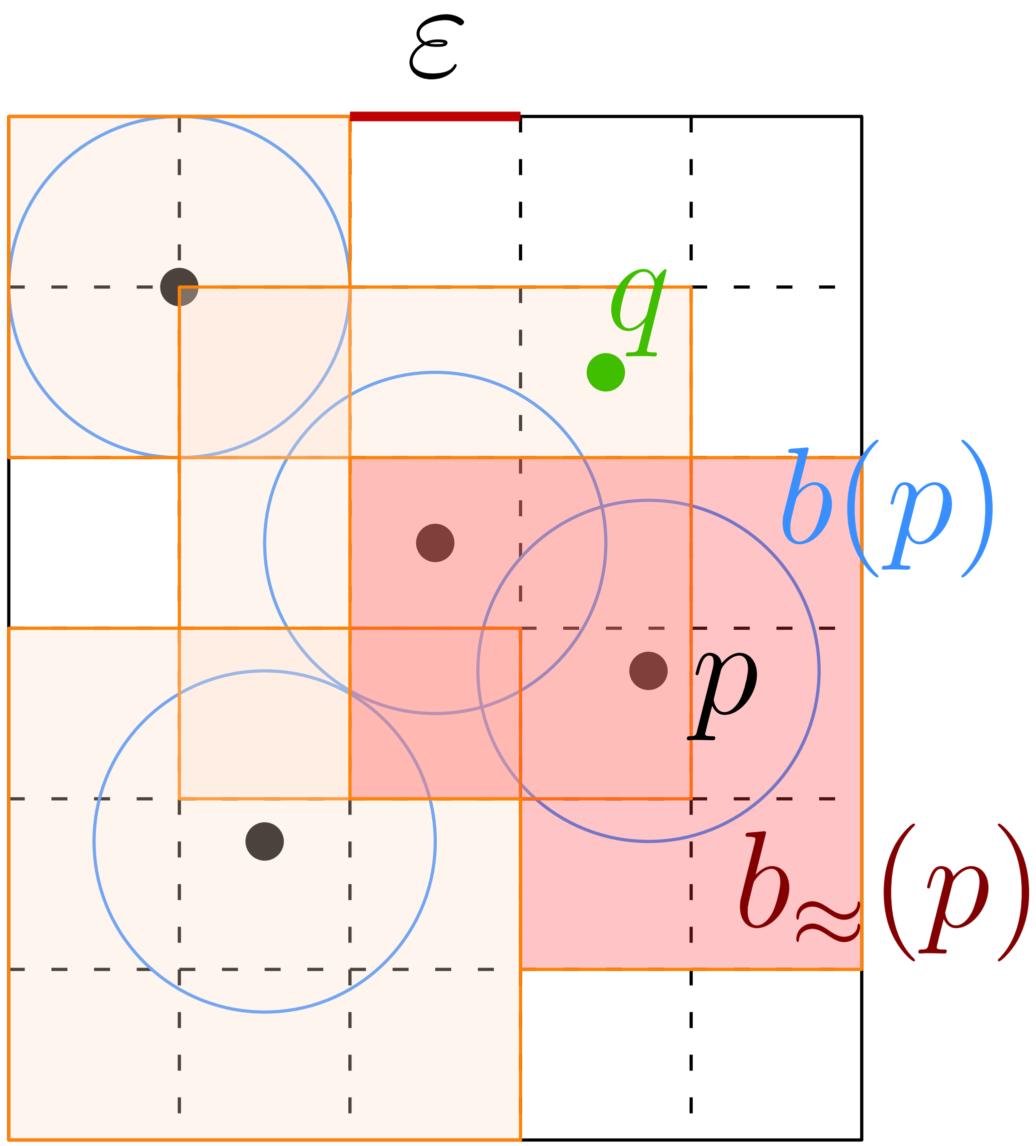
Approximating the ball

- Divide the space into a grid with sides ε
- Define $b_{\approx}(p)$ as the grid cells intersected by $b(p)$
- Throw all $b_{\approx}$ into a hashtable
- Now deciding whether point q falls into a certain range is easy: $O(1)$



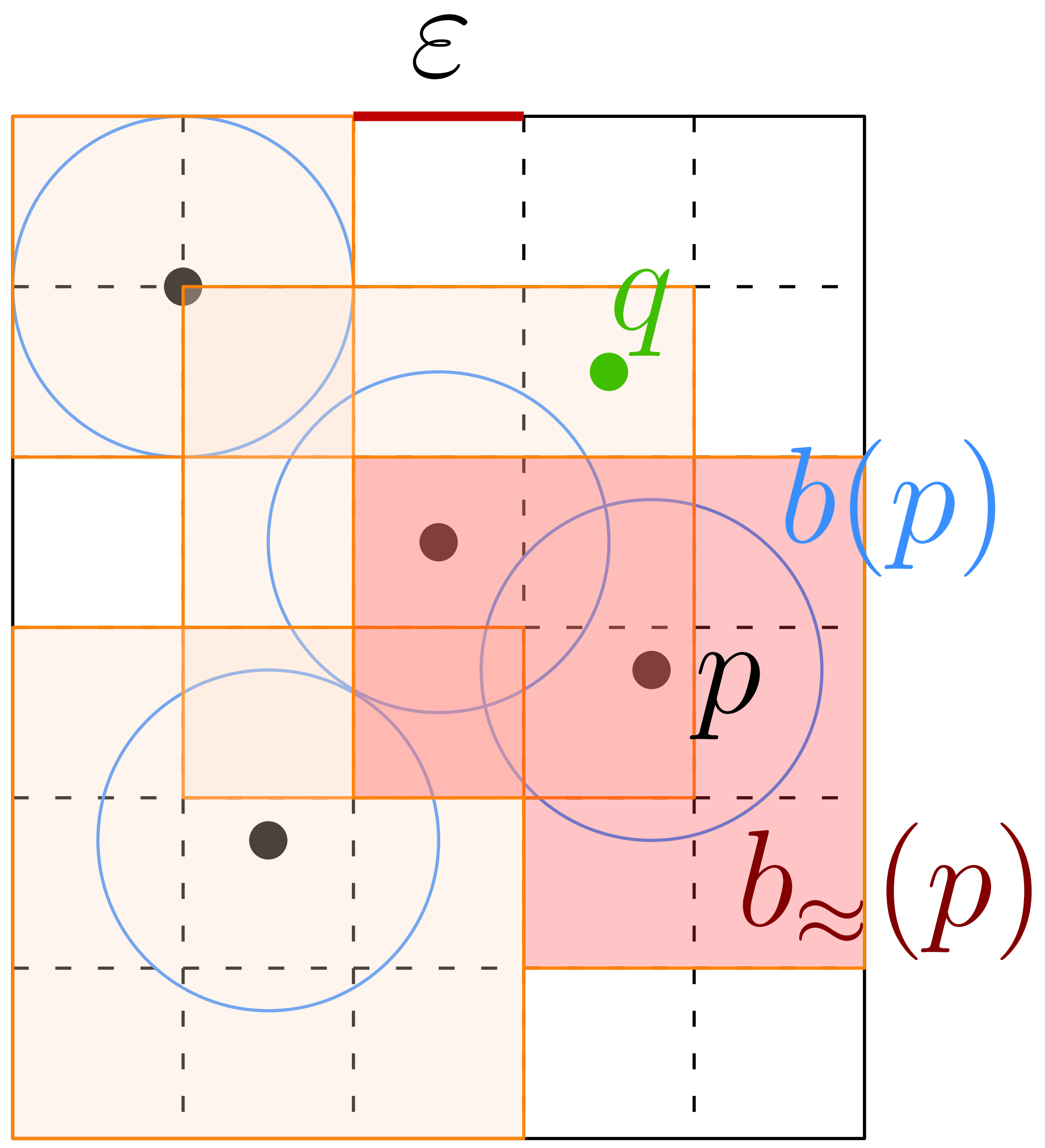
Approximating the ball

- Divide the space into a grid with sides ε
- Define $b_{\approx}(p)$ as the grid cells intersected by $b(p)$
- Throw all $b_{\approx}$ into a hashtable
- Now deciding whether point q falls into a certain range is easy: $O(1)$
- For constant ball size this only takes $O(n/\varepsilon^d)$ space!



Approximating the ball

- Divide the space into a grid with sides ε
- Define $b_{\approx}(p)$ as the grid cells intersected by $b(p)$
- Throw all $b_{\approx}$ into a hashtable
- Now deciding whether point q falls into a certain range is easy: $O(1)$
- For constant ball size this only takes $O(n/\varepsilon^d)$ space!



But we don't have constant ball sizes…

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1 + \varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$

For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

p is only returned if there are two consecutive indices i and $i+1$ such that q is in the ball set of $i+1$ but not in the ball set of i

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

p is only returned if there are two consecutive indices i and $i+1$ such that q is in the ball set of $i+1$ but not in the ball set of i

$$r(1+\varepsilon/16)^i \leq \mathbf{d}(q, P) \leq \mathbf{d}(q, p) \leq r(1+\varepsilon/16)^{i+1}(1+\varepsilon/16) \leq (1+\varepsilon/16)^2\mathbf{d}(q, P) \leq (1+\varepsilon/4)\mathbf{d}(q, P)$$

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

p is only returned if there are two consecutive indices i and $i+1$ such that q is in the ball set of $i+1$ but not in the ball set of i

$$r(1+\varepsilon/16)^i \leq \mathbf{d}(q,P) \leq \mathbf{d}(q,p) \leq \boxed{\boxed{r(1+\varepsilon/16)^{i+1}}\boxed{(1+\varepsilon/16)}} \leq (1+\varepsilon/16)^2\mathbf{d}(q,P) \leq (1+\varepsilon/4)\mathbf{d}(q,P)$$

Approximation from using balls

Approximation from approximating the balls

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

p is only returned if there are two consecutive indices i and $i+1$ such that q is in the ball set of $i+1$ but not in the ball set of i

$$r(1+\varepsilon/16)^i \leq \boxed{\mathbf{d}(q,P)} \leq \mathbf{d}(q,p) \leq r(1+\varepsilon/16)^{i+1}(1+\varepsilon/16) \leq \boxed{(1+\varepsilon/16)^2\mathbf{d}(q,P)} \leq (1+\varepsilon/4)\mathbf{d}(q,P)$$

Substitute

Approximate interval structure

Lemma Let $\mathcal{I}_{\approx}(P, r, R, \varepsilon/16)$ be a $(1+\varepsilon/16)$-approximation of $\mathcal{I}(P, r, R, \varepsilon/16)$
For a query point $q \in \mathcal{M}$ if $\mathcal{I}_{\approx}$ returns a target set that is an approximation of a ball in $\mathcal{I}$ centered at a point p with radius $\alpha \in [r, R]$ then p is a $(1+\varepsilon/4)$-ANN to q

Proof:

p is only returned if there are two consecutive indices i and $i+1$ such that q is in the ball set of $i+1$ but not in the ball set of i

$$r(1+\varepsilon/16)^i \le \mathbf{d}(q, P) \le \mathbf{d}(q, p) \le r(1+\varepsilon/16)^{i+1}(1+\varepsilon/16) \le$$
$$\boxed{(1+\varepsilon/16)^2 \mathbf{d}(q, P) \le (1+\varepsilon/4)\mathbf{d}(q, P)}$$

$$\boxed{1 + \frac{2\varepsilon}{16} + \frac{\varepsilon^2}{16^2} = 1 + \frac{\varepsilon}{8} + \frac{\varepsilon}{16} = 1 + \frac{3\varepsilon}{16} \le 1 + \frac{4}{\varepsilon}}$$

Intermediate results

Intermediate results

- Given a set P of n points in $\mathbb{R}^d$, one can compute a set of $\mathcal{B}$ of $O(\frac{n}{\varepsilon}\log n)$ balls

Intermediate results

- Given a set P of n points in $\mathbb{R}^d$, one can compute a set of $\mathcal{B}$ of $O(\frac{n}{\varepsilon}\log n)$ balls
- s.t. answering $(1+\varepsilon)$-ANN queries on P can be answered by doing a single target query on $\mathcal{B}$

Intermediate results

- Given a set P of n points in $\mathbb{R}^d$, one can compute a set of $\mathcal{B}$ of $O(\frac{n}{\varepsilon}\log n)$ balls
- s.t. answering $(1+\varepsilon)$-ANN queries on P can be answered by doing a single target query on $\mathcal{B}$
- Furthermore, if we $(1+\varepsilon/16)$-approximate each ball the target query becomes easier.

Improvements in low dimensions (for large ε)

Improvements in low dimensions (for large ε)

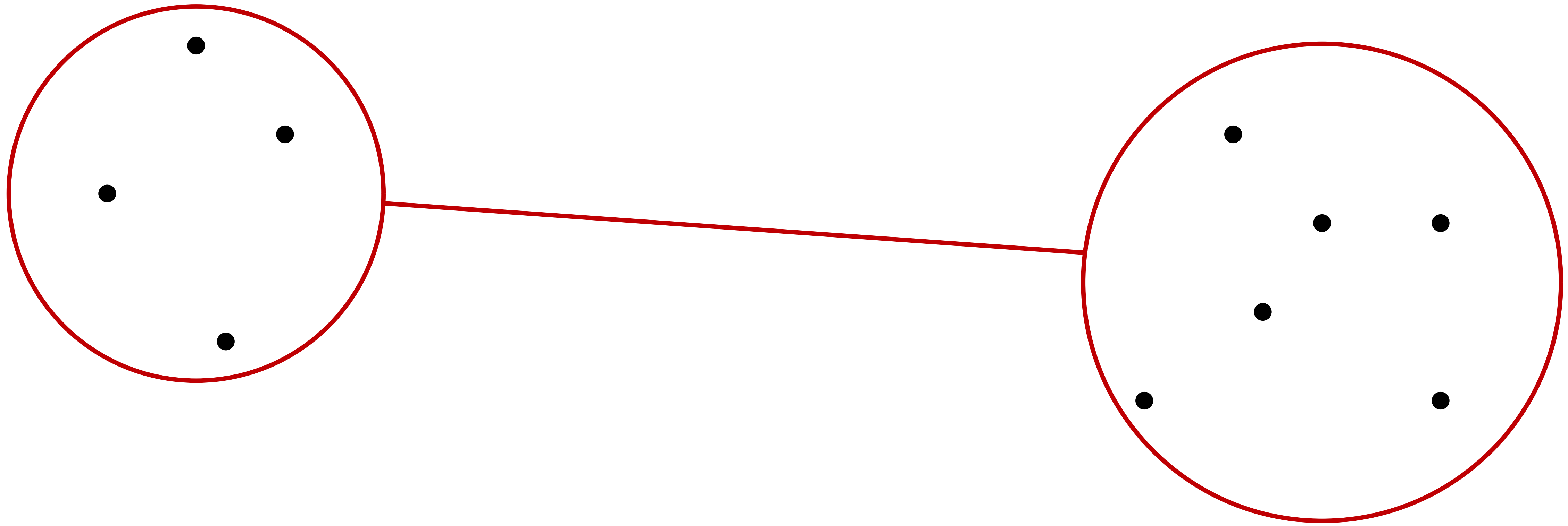
- Initial, simple construction (previous lecture): balls per pair of points

Improvements in low dimensions (for large ε)

- Initial, simple construction (previous lecture): balls per pair of points
- How can we reduce the number of pairs?

Improvements in low dimensions (for large ε)

- Initial, simple construction (previous lecture): balls per pair of points
- How can we reduce the number of pairs?
- Well Separated Pair Decomposition!



Improvements in low dimensions

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u, v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u, v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

$$\mathcal{B}(rep_u, rep_v) = \{\mathbf{b}(rep_u, r), \mathbf{b}(rep_v, r) | r = (1+\varepsilon/3)^i \in \mathcal{J}(u, v)\}$$

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u,v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

$$\mathcal{B}(rep_u, rep_v) = \{\mathbf{b}(rep_u, r), \mathbf{b}(rep_v, r) | r = (1+\varepsilon/3)^i \in \mathcal{J}(u,v)\}$$

and

$$\mathcal{J}(u,v) = [\tfrac{1}{8}, \tfrac{4}{\varepsilon}] \cdot \|rep_u - rep_v\|$$

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u, v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

$$\mathcal{B}(rep_u, rep_v) = \{\mathbf{b}(rep_u, r), \mathbf{b}(rep_v, r) | r = (1+\varepsilon/3)^i \in \mathcal{J}(u,v)\}$$

and

$$\mathcal{J}(u,v) = [\tfrac{1}{8}, \tfrac{4}{\varepsilon}] \cdot \|rep_u - rep_v\|$$

- We have $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ balls per pair

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u, v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

$$\mathcal{B}(rep_u, rep_v) = \{\mathbf{b}(rep_u, r), \mathbf{b}(rep_v, r) | r = (1+\varepsilon/3)^i \in \mathcal{J}(u,v)\}$$

and

$$\mathcal{J}(u,v) = [\tfrac{1}{8}, \tfrac{4}{\varepsilon}] \cdot \|rep_u - rep_v\|$$

- We have $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ balls per pair
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$

Improvements in low dimensions

- Construct a (c/ε)-WSPD $\mathcal{W}$ of P, where c is sufficiently large
- The number of pairs in a WSPD is $O(\frac{n}{\varepsilon^d})$
- For every pair $\{u, v\} \in \mathcal{W}$ compute $\mathcal{B}(rep_u, rep_v)$ and add it to $\mathcal{B}$ where:

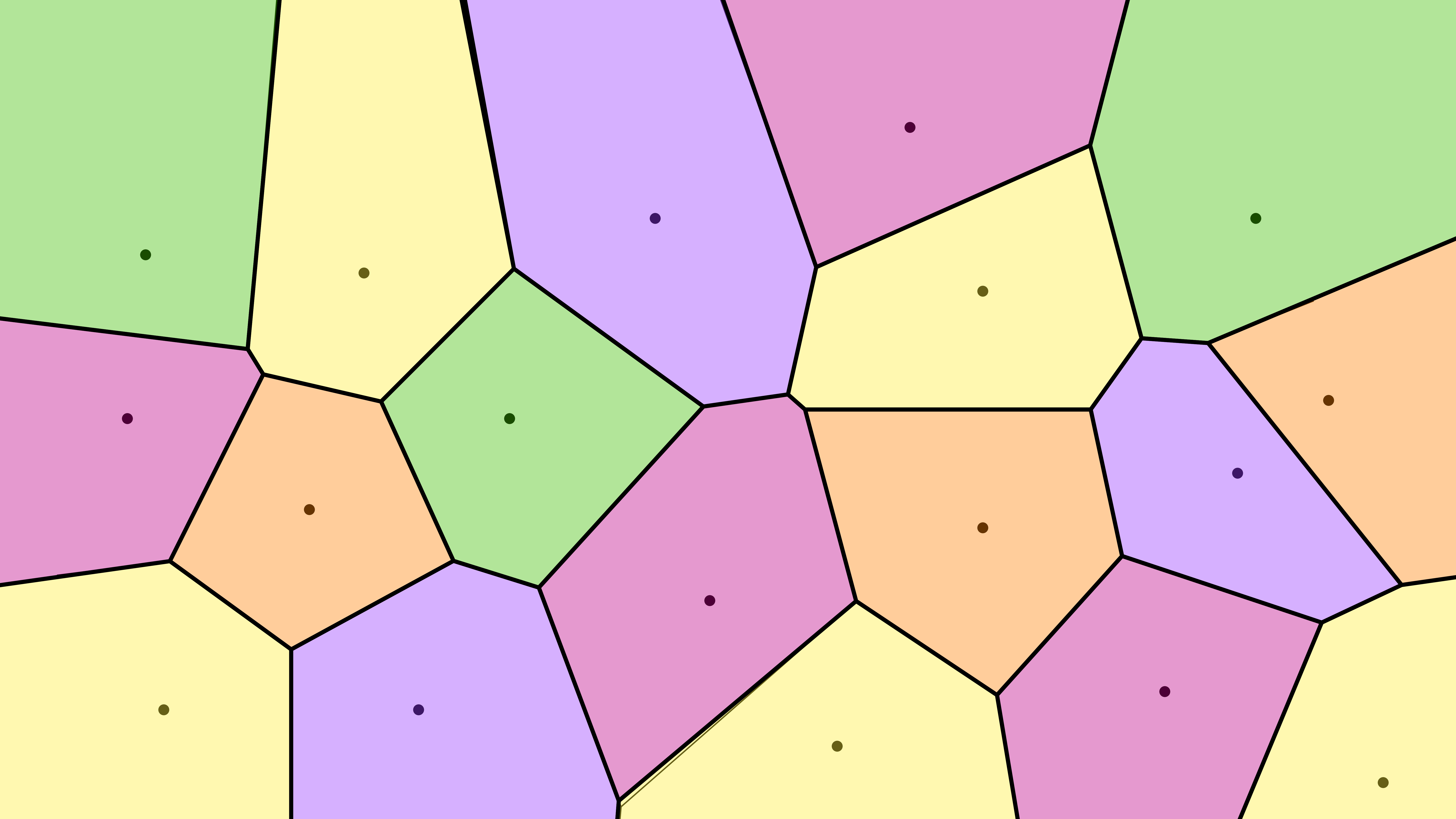
$$\mathcal{B}(rep_u, rep_v) = \{\mathbf{b}(rep_u, r), \mathbf{b}(rep_v, r) | r = (1 + \varepsilon/3)^i \in \mathcal{J}(u, v)\}$$

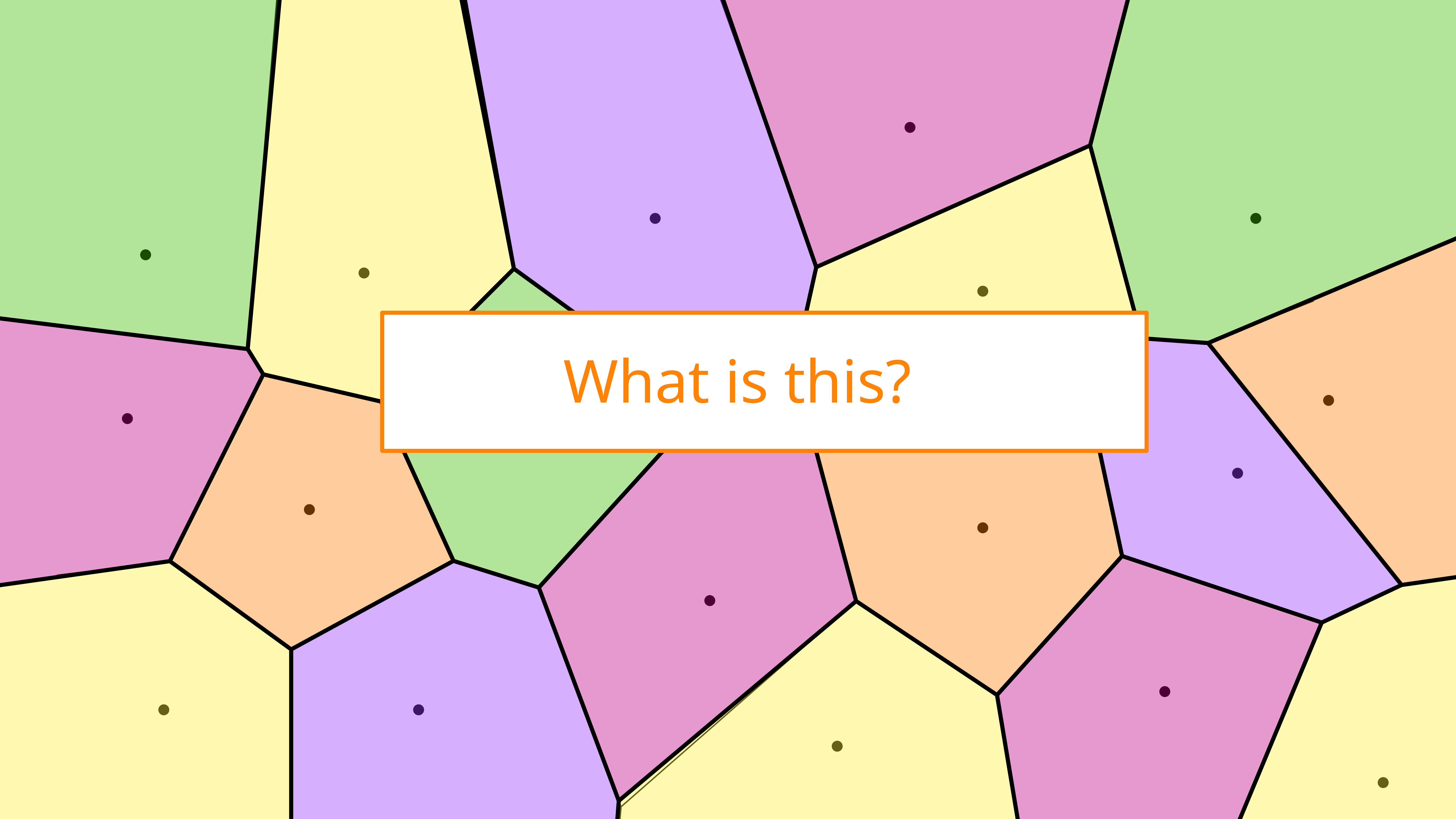
and

$$\mathcal{J}(u, v) = [\tfrac{1}{8}, \tfrac{4}{\varepsilon}] \cdot \|rep_u - rep_v\|$$

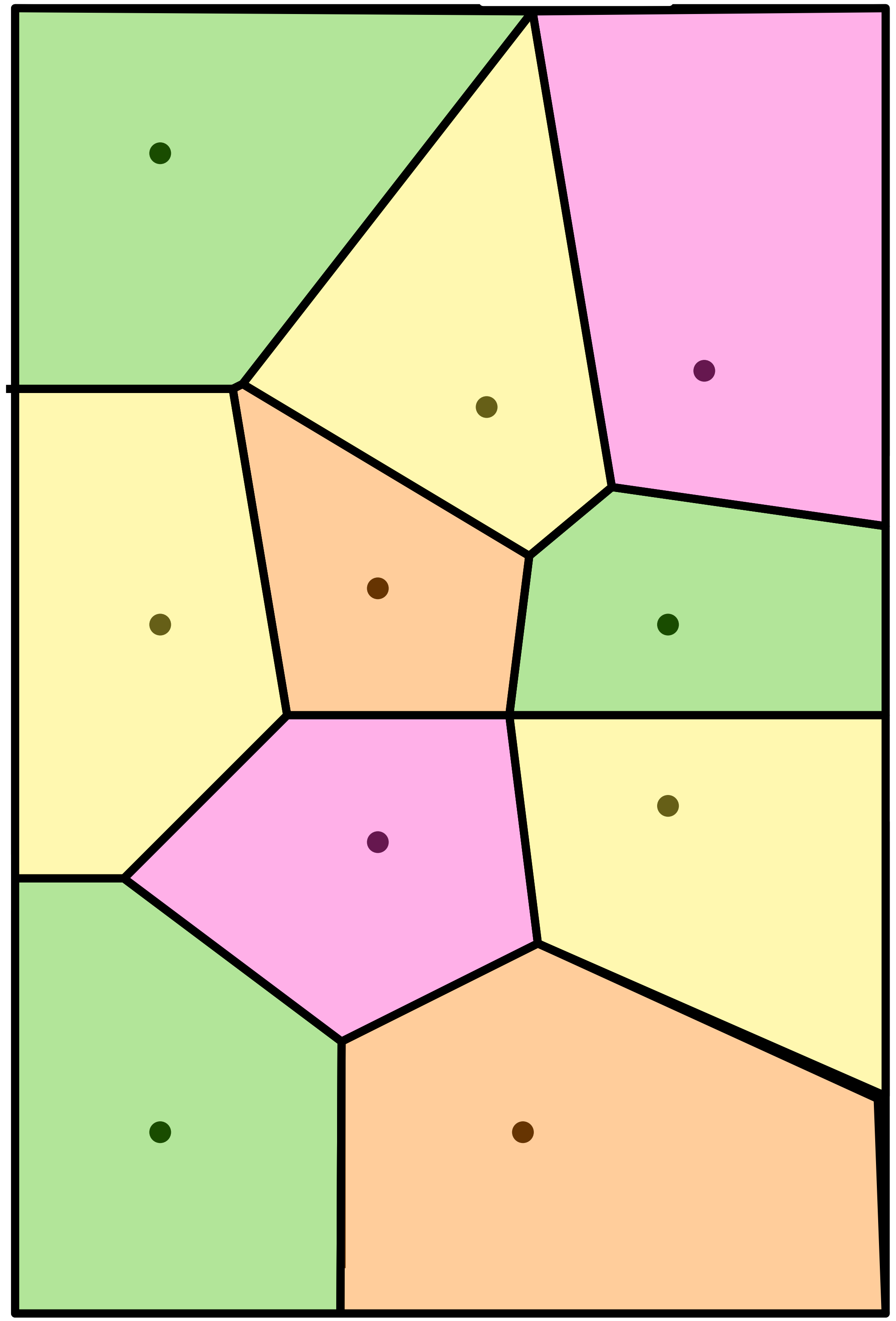
- We have $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ balls per pair
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$

Correctness proof: as exercise



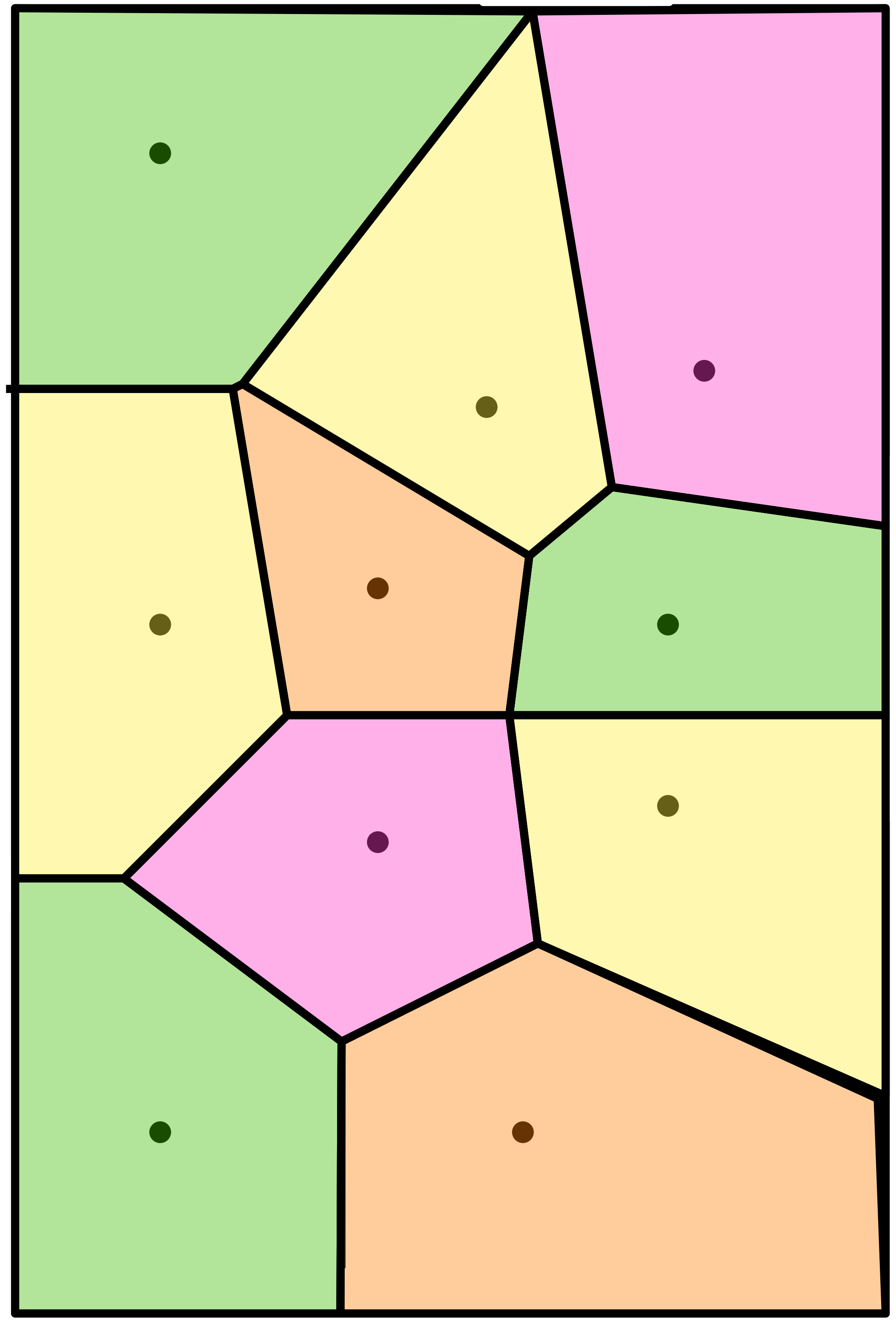

What is this?

Motivation



Motivation

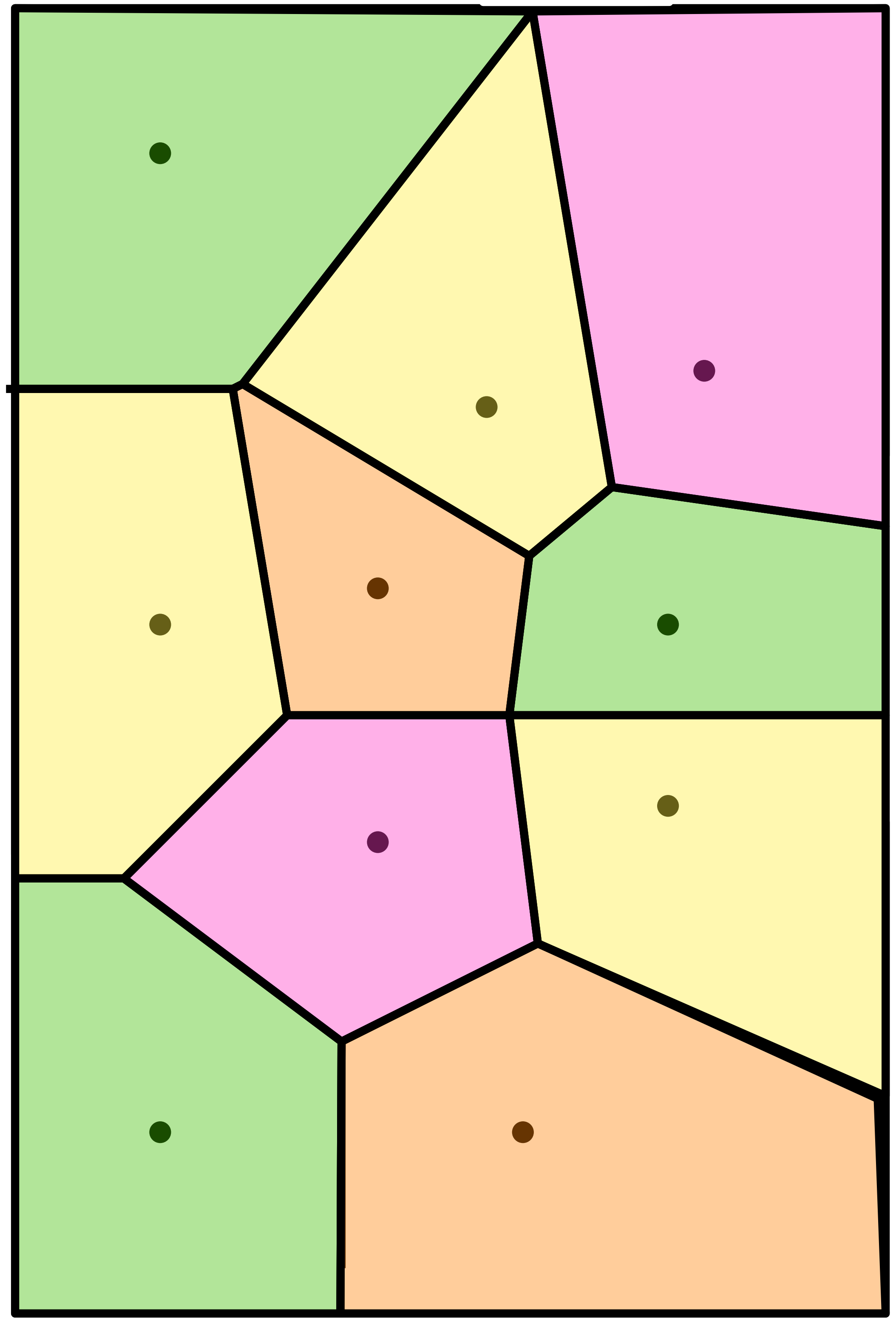
Voronoi diagrams have a multitude of uses:



Motivation

Voronoi diagrams have a multitude of uses:

- *Biology* Model biological structures like cells
- *Hydrology* Calculate the rainfall in an area based on point measurements
- *Aviation* Find the nearest safe landing zone in case of failure



What is a Voronoi Diagram?

What is a Voronoi Diagram?

A Voronoi diagram V of a point set $P \subseteq \mathbb{R}^d$ is a partition of space into regions such that a cell of point $p \in P$ is:

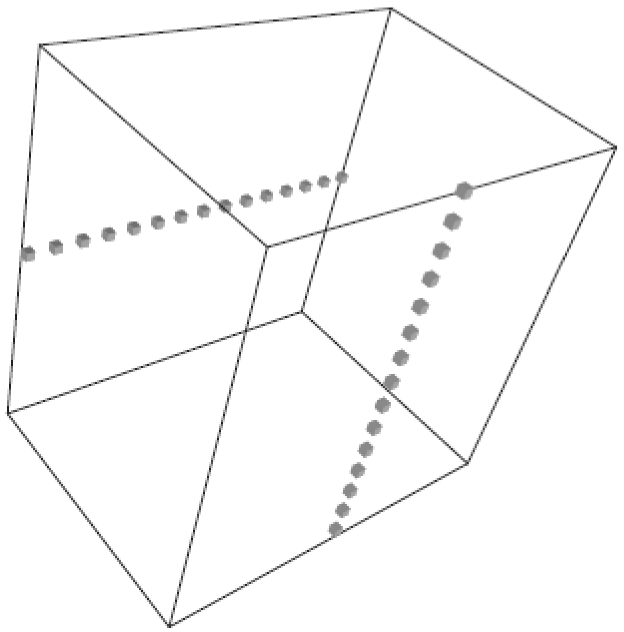
$$V(p, P) = s \in \mathbb{R}^d | \|s - p\| \leq \|s - p'\| \textbf{for all}\, p' \in P$$

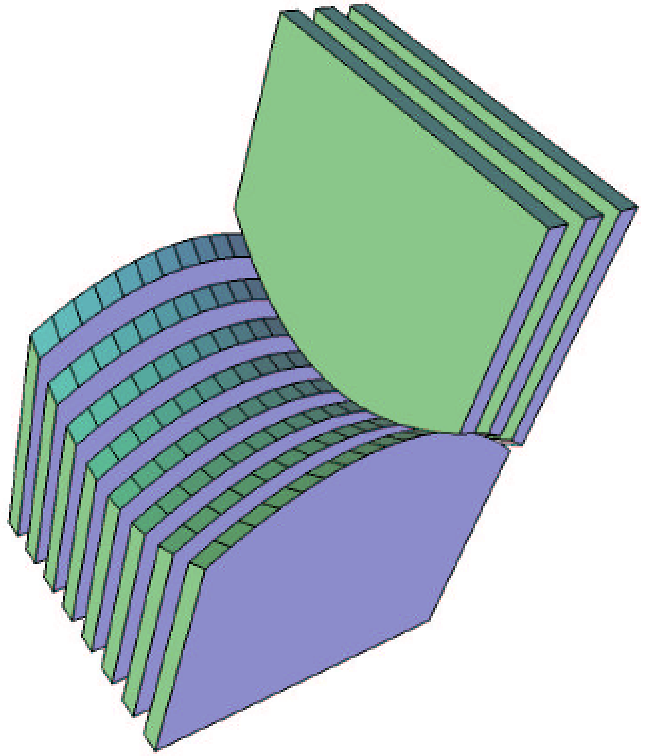
What is a Voronoi Diagram?

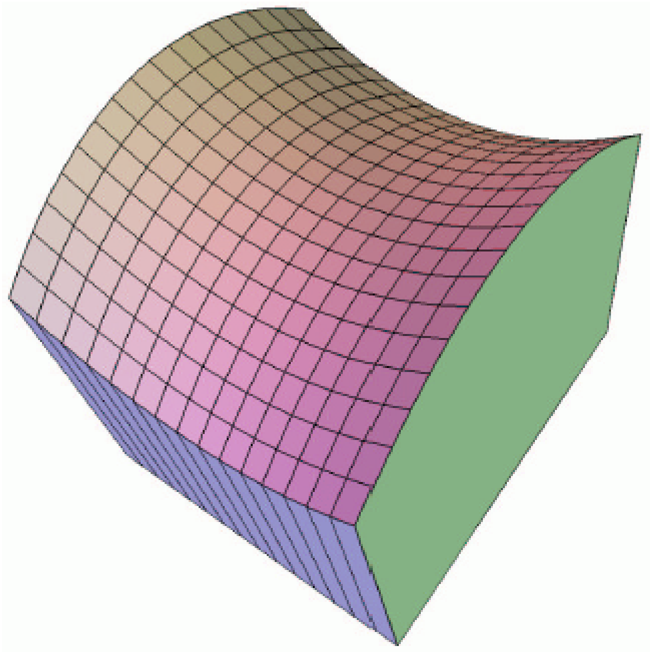
A Voronoi diagram V of a point set $P \subseteq \mathbb{R}^d$ is a partition of space into regions such that a cell of point $p \in P$ is:

$$V(p, P) = s \in \mathbb{R}^d | \|s - p\| \leq \|s - p'\| \textbf{for all } p' \in P$$

However, it has complexity $O(n^{\lceil \frac{d}{2} \rceil})$ in $\mathbb{R}^d$ in the worst case







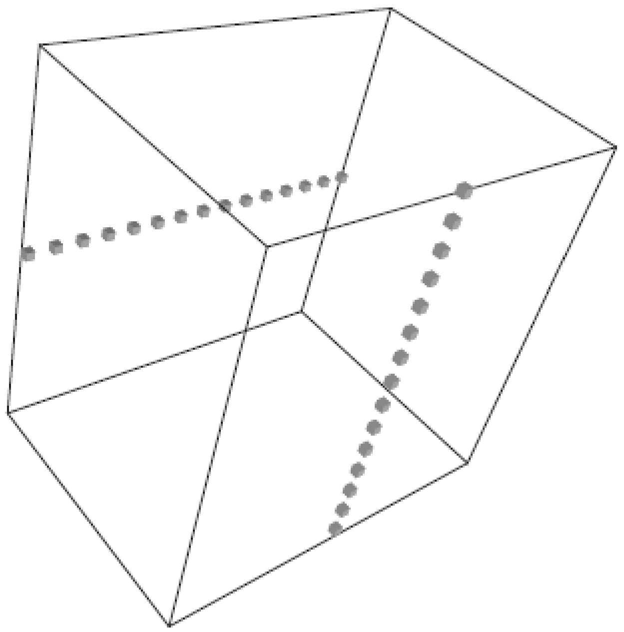
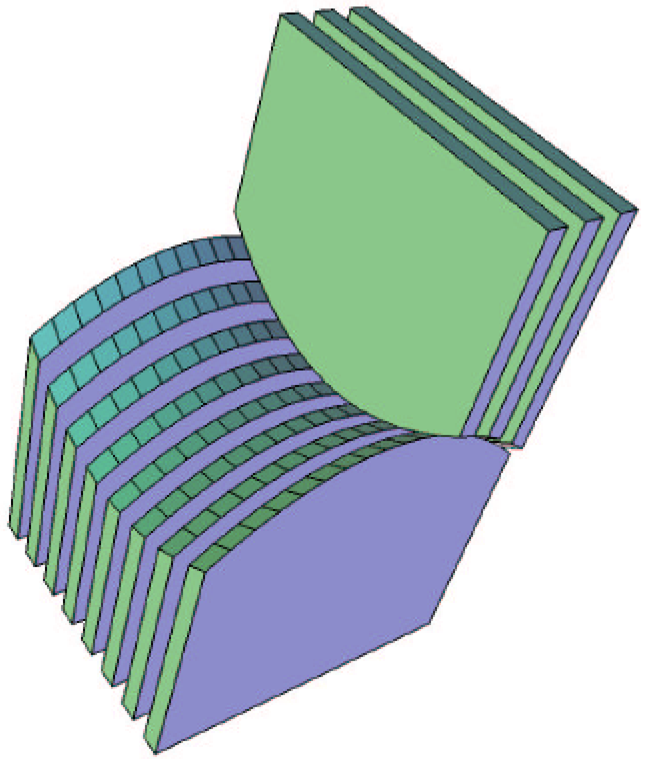
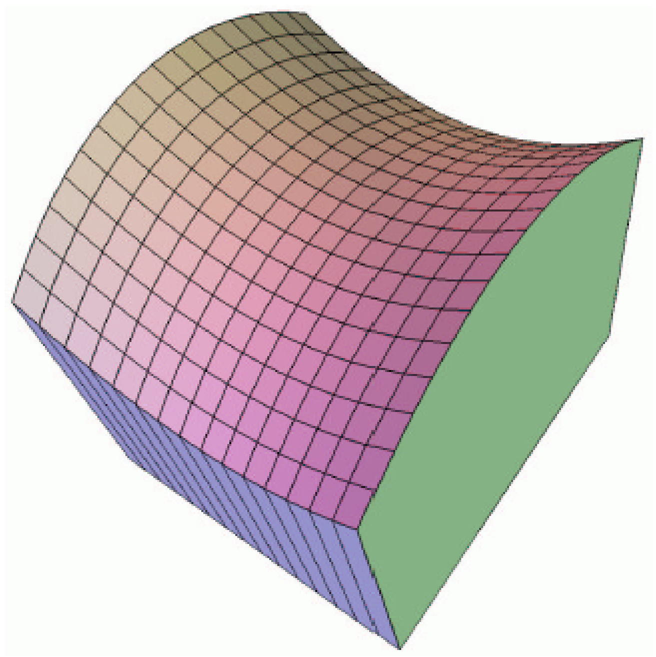
What is a Voronoi Diagram?

A Voronoi diagram V of a point set $P \subseteq \mathbb{R}^d$ is a partition of space into regions such that a cell of point $p \in P$ is:

$$V(p, P) = s \in \mathbb{R}^d | \|s - p\| \leq \|s - p'\| \textbf{for all } p' \in P$$

However, it has complexity $O(n^{\lceil \frac{d}{2} \rceil})$ in $\mathbb{R}^d$ in the worst case

Can we do better?

Approximate Voronoi diagrams

Approximate Voronoi diagrams

Definition: **Approximate Voronoi Diagram**

Given a set P of n points in $\mathbb{R}^d$ and parameter $\varepsilon > 0$, a $(1+\varepsilon)$-*Approximated Voronoi Diagram(AVS)* of P is a partition $\mathcal{V}$ of $\mathbb{R}^d$ into regions φ, s.t. for any region $\varphi \in \mathcal{V}$ we have that rep_φ is a $(1+\varepsilon)$-ANN for x, that is:

Approximate Voronoi diagrams

Definition: **Approximate Voronoi Diagram**
Given a set P of n points in $\mathbb{R}^d$ and parameter $\varepsilon > 0$, a $(1+\varepsilon)$-*Approximated Voronoi Diagram(AVS)* of P is a partition $\mathcal{V}$ of $\mathbb{R}^d$ into regions φ, s.t. for any region $\varphi \in \mathcal{V}$ we have that rep_φ is a $(1+\varepsilon)$-ANN for x, that is:

$$\forall x \in \varphi \|x - rep_\varphi\| \leq (1+\varepsilon) d(x, P)$$

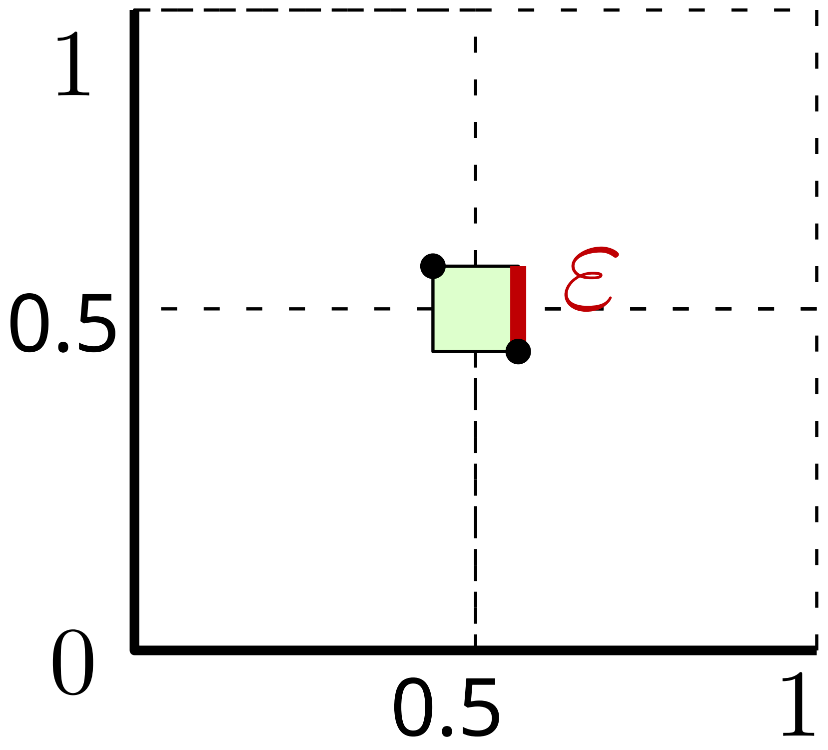
Approximate Nearest Neighbors in $\mathbb{R}^d$

Approximate Nearest Neighbors in $\mathbb{R}^d$

(now fast, using approximate Voronoi diagrams)

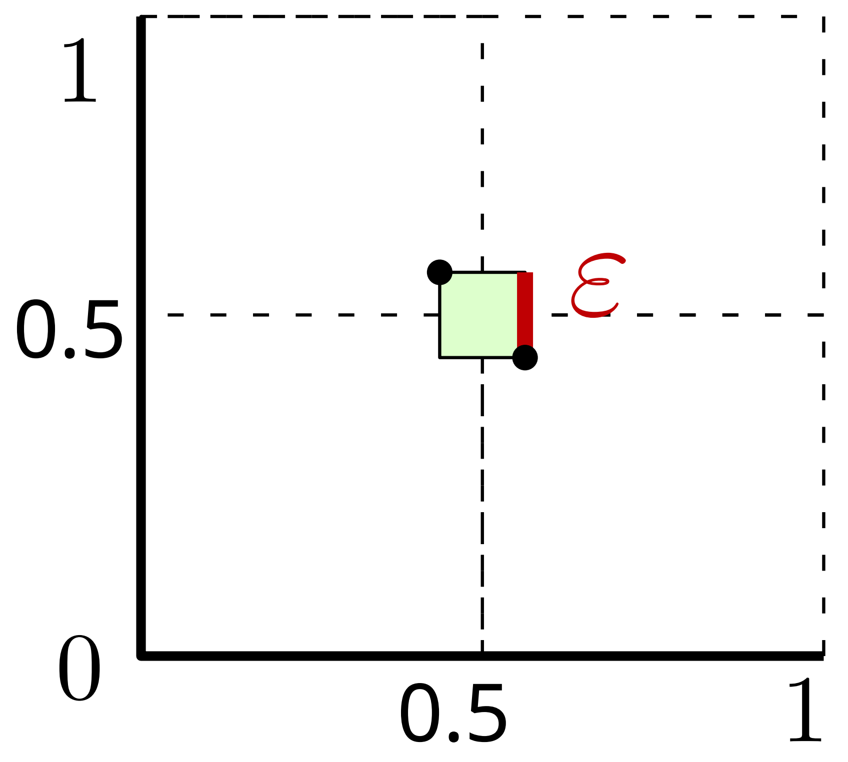
Fast ANN in $\mathbb{R}^d$

- In the following, asssume P is a set of points contained in hypercube $[0.5 - \varepsilon/d, 0.5 + \varepsilon/d]^d$



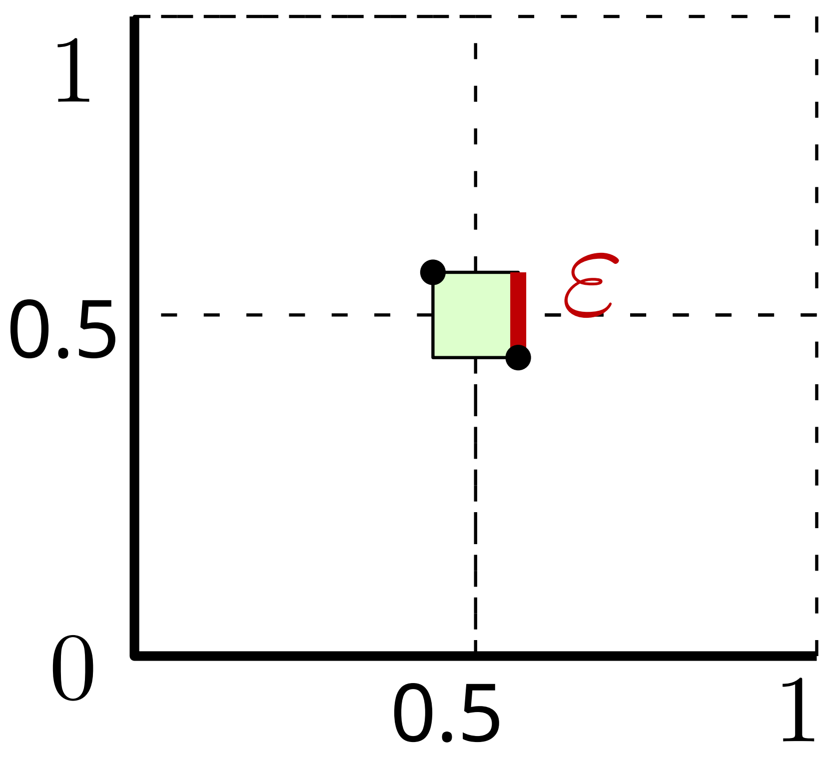
Fast ANN in $\mathbb{R}^d$

- In the following, asssume P is a set of points contained in hypercube $[0.5 - \varepsilon/d, 0.5 + \varepsilon/d]^d$
- Guarantee by some transformation T



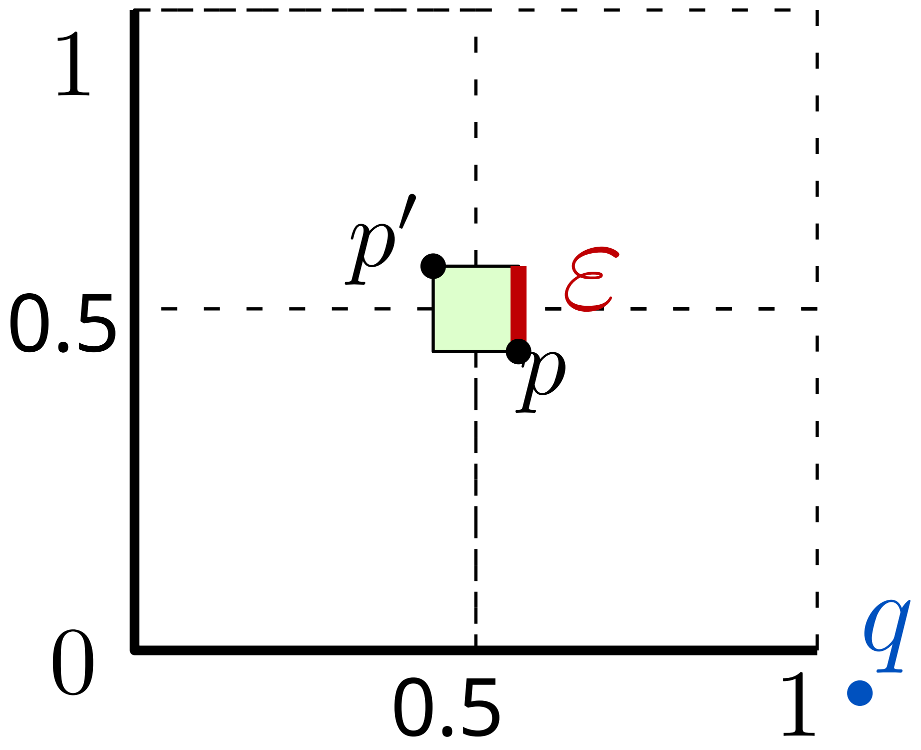
Fast ANN in $\mathbb{R}^d$

- In the following, asssume P is a set of points contained in hypercube $[0.5 - \varepsilon/d, 0.5 + \varepsilon/d]^d$
- Guarantee by some transformation T
- Computing ANN of q on P is equivalent to computing the ANN of $T(q)$ on $T(P)$



Fast ANN in $\mathbb{R}^d$

- In the following, asssume P is a set of points contained in hypercube $[0.5 - \varepsilon/d, 0.5 + \varepsilon/d]^d$
- Guarantee by some transformation T
- Computing ANN of q on P is equivalent to computing the ANN of $T(q)$ on $T(P)$
- If q is outside the unit hypercube $[0,1]^d$ any $p \in P$ is an $(1+\varepsilon)$-ANN

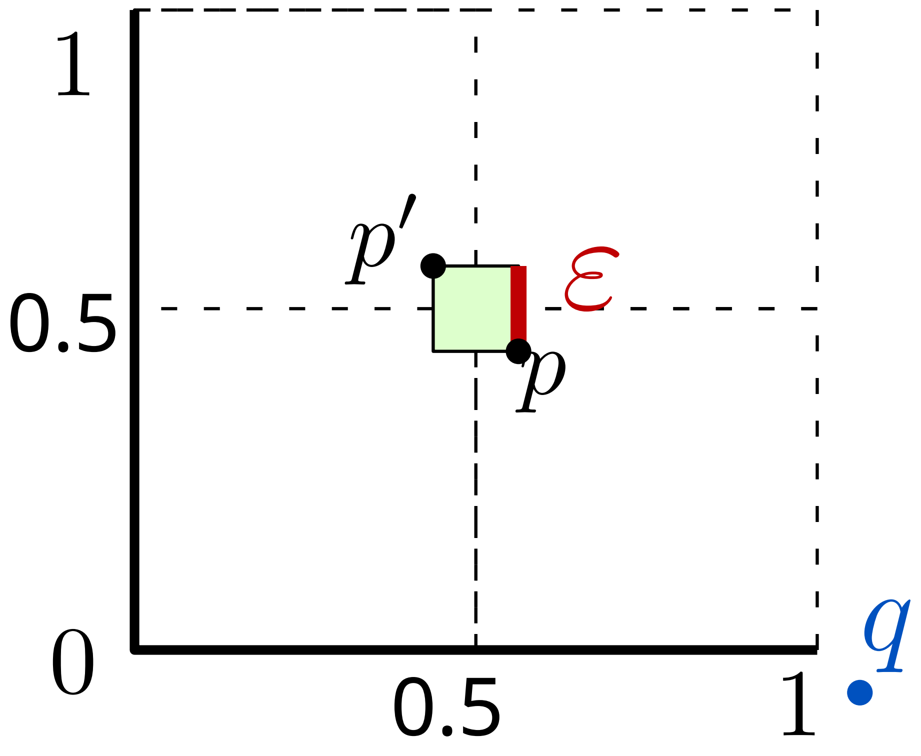


(Exercise: Check, in doubt change constants)

Fast ANN in $\mathbb{R}^d$

- In the following, asssume P is a set of points contained in hypercube $[0.5 - \varepsilon/d, 0.5 + \varepsilon/d]^d$
- Guarantee by some transformation T
- Computing ANN of q on P is equivalent to computing the ANN of $T(q)$ on $T(P)$
- If q is outside the unit hypercube $[0,1]^d$ any $p \in P$ is an $(1+\varepsilon)$-ANN

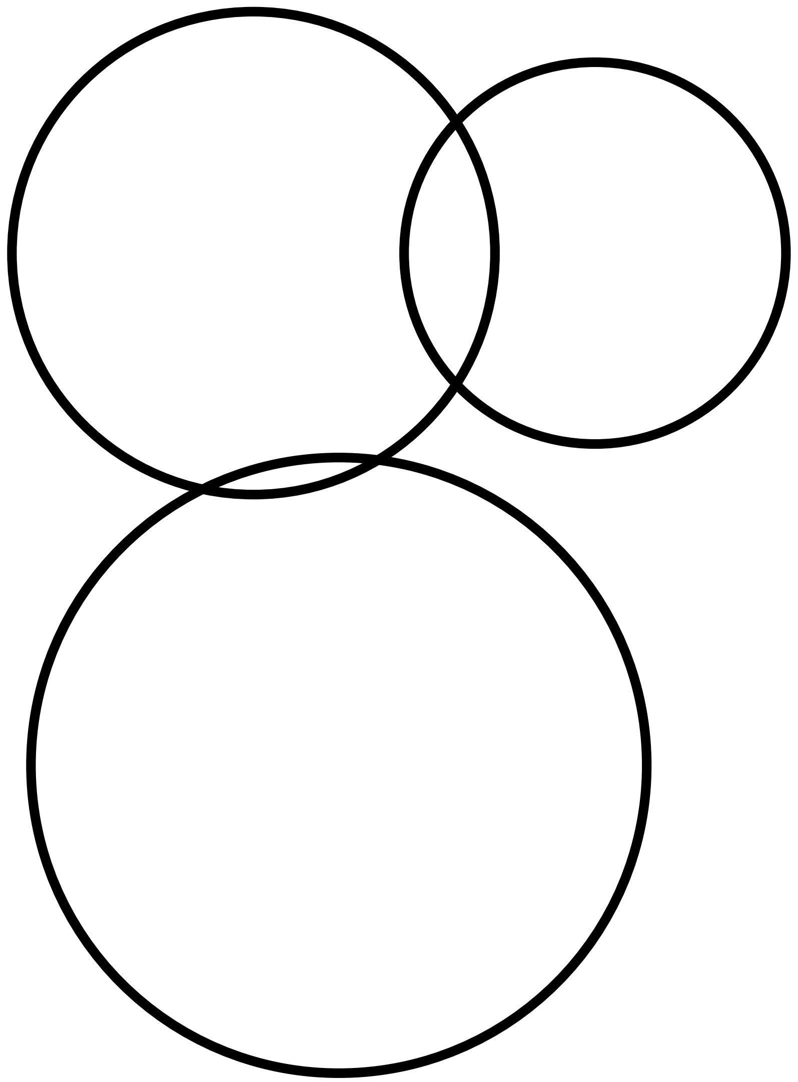
(Exercise: Check, in doubt change constants)



Thus only consider ANN for points inside $[0,1]^d$

Creating the AVD

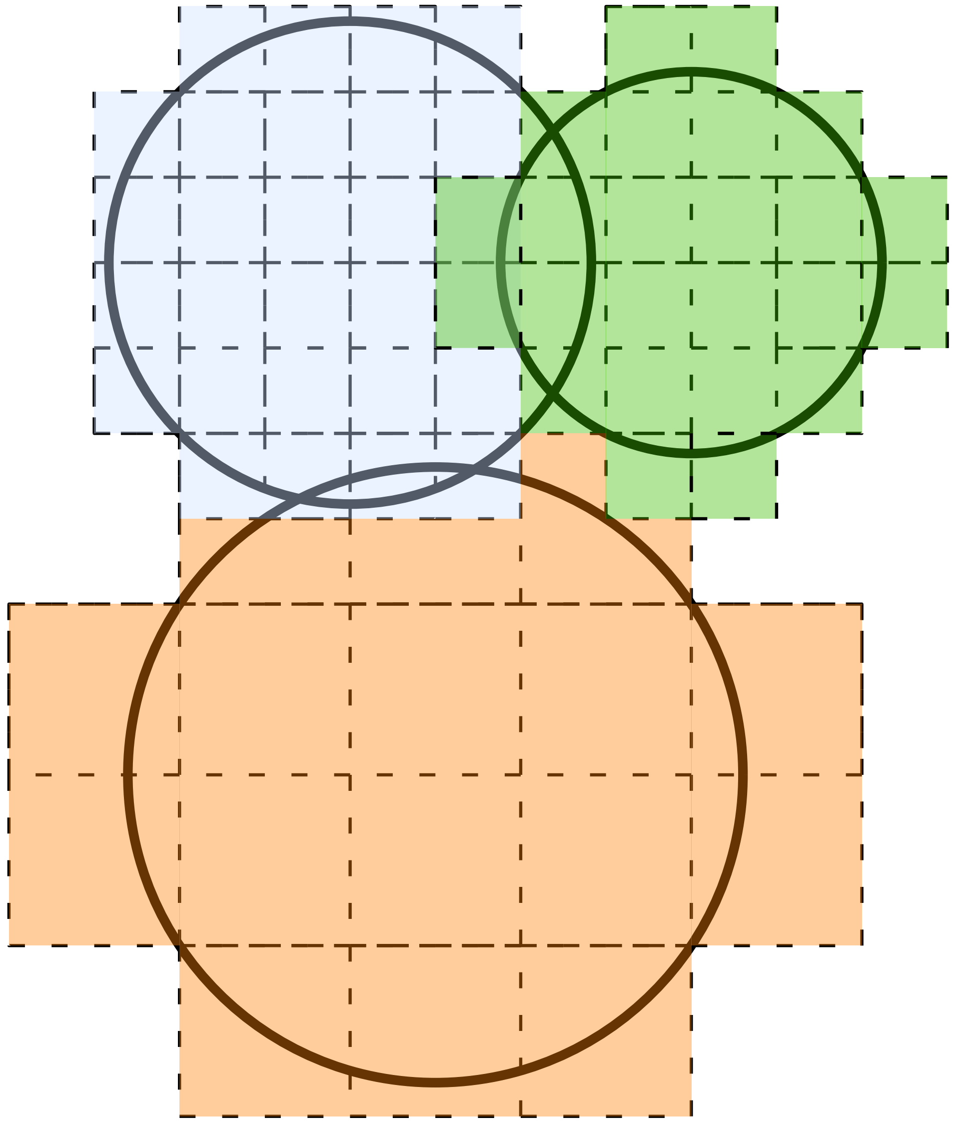
- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls



Creating the AVD

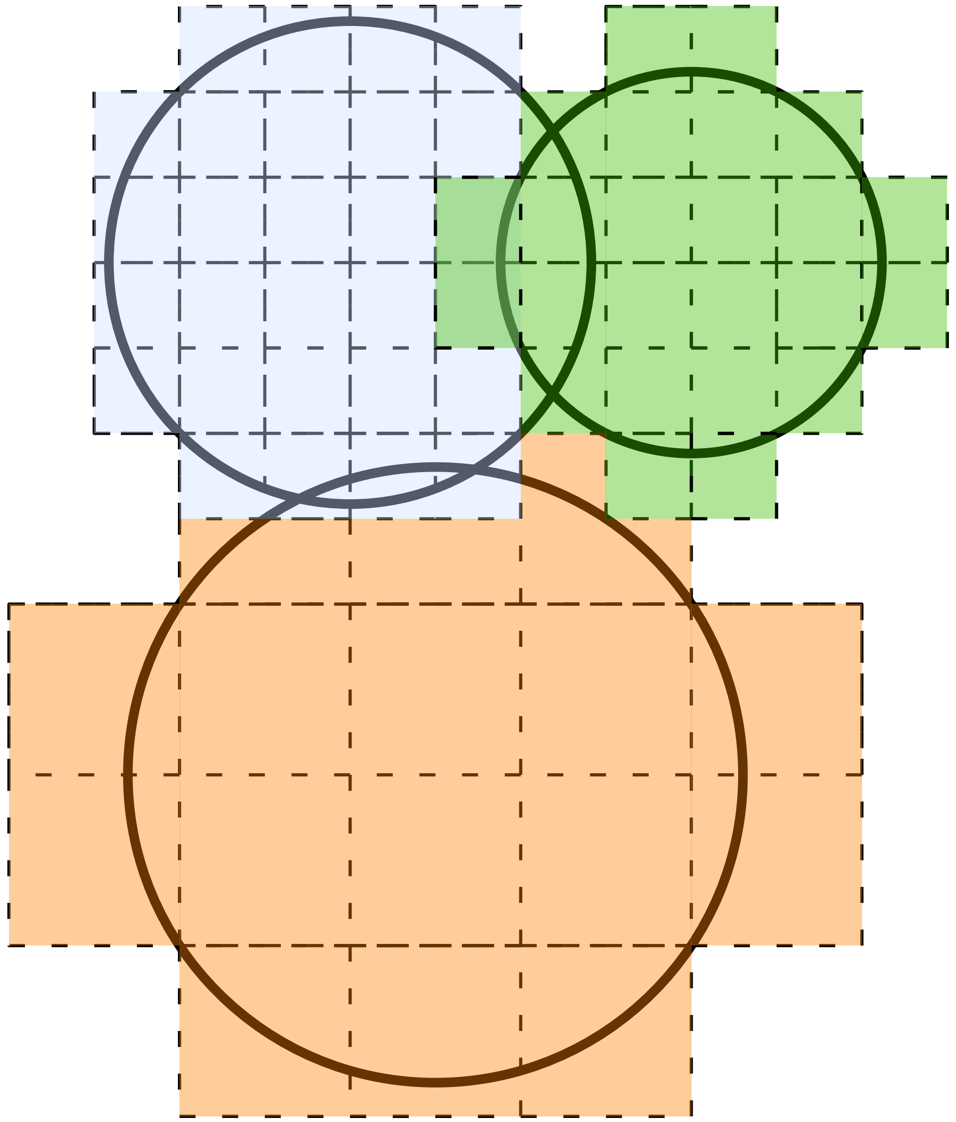
- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls

- Approximate b by the cells $\mathcal{C}'$ that intersect it



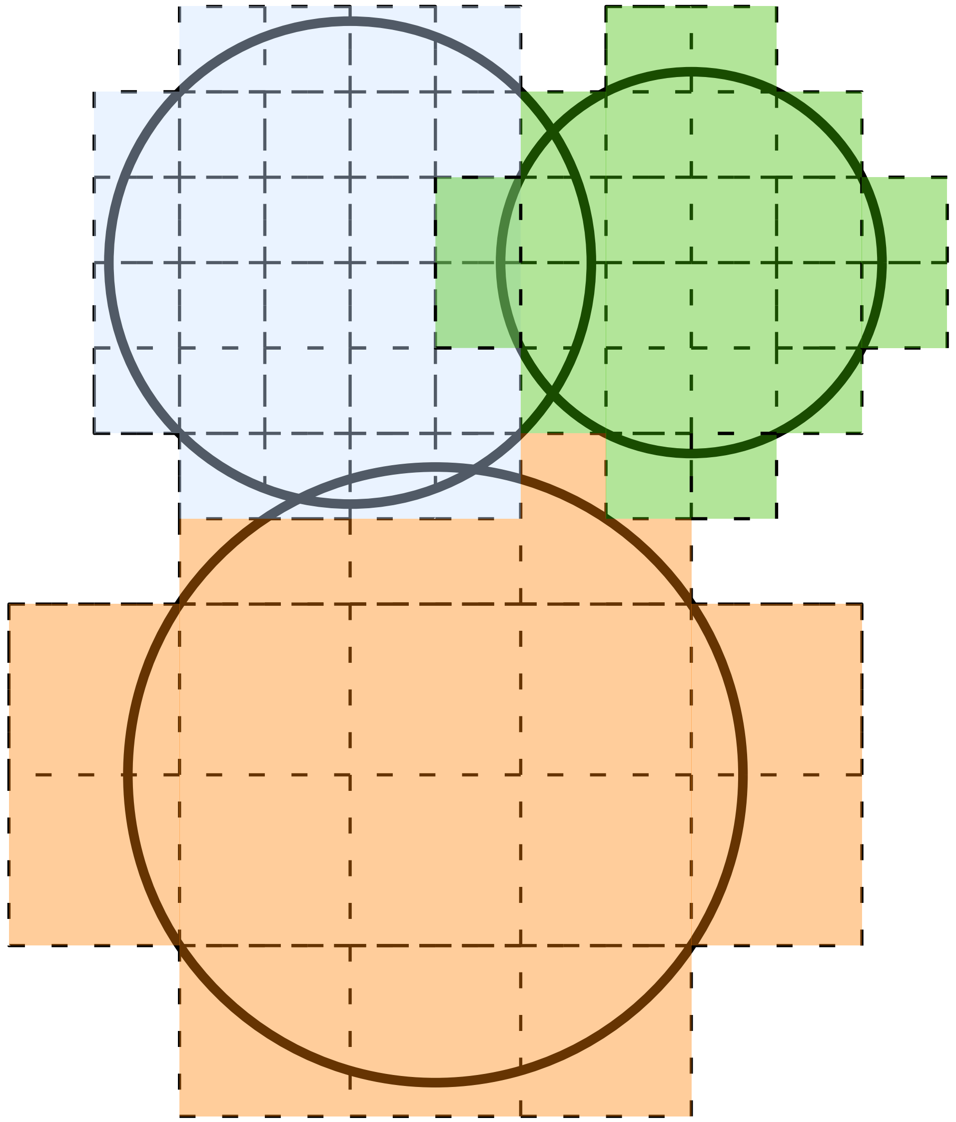
Creating the AVD

- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls
- Approximate b by the cells $\mathcal{C}'$ that intersect it
- Pick grid G_{2^i} s.t. $\sqrt{d}2^i \leq (\varepsilon/16)r$



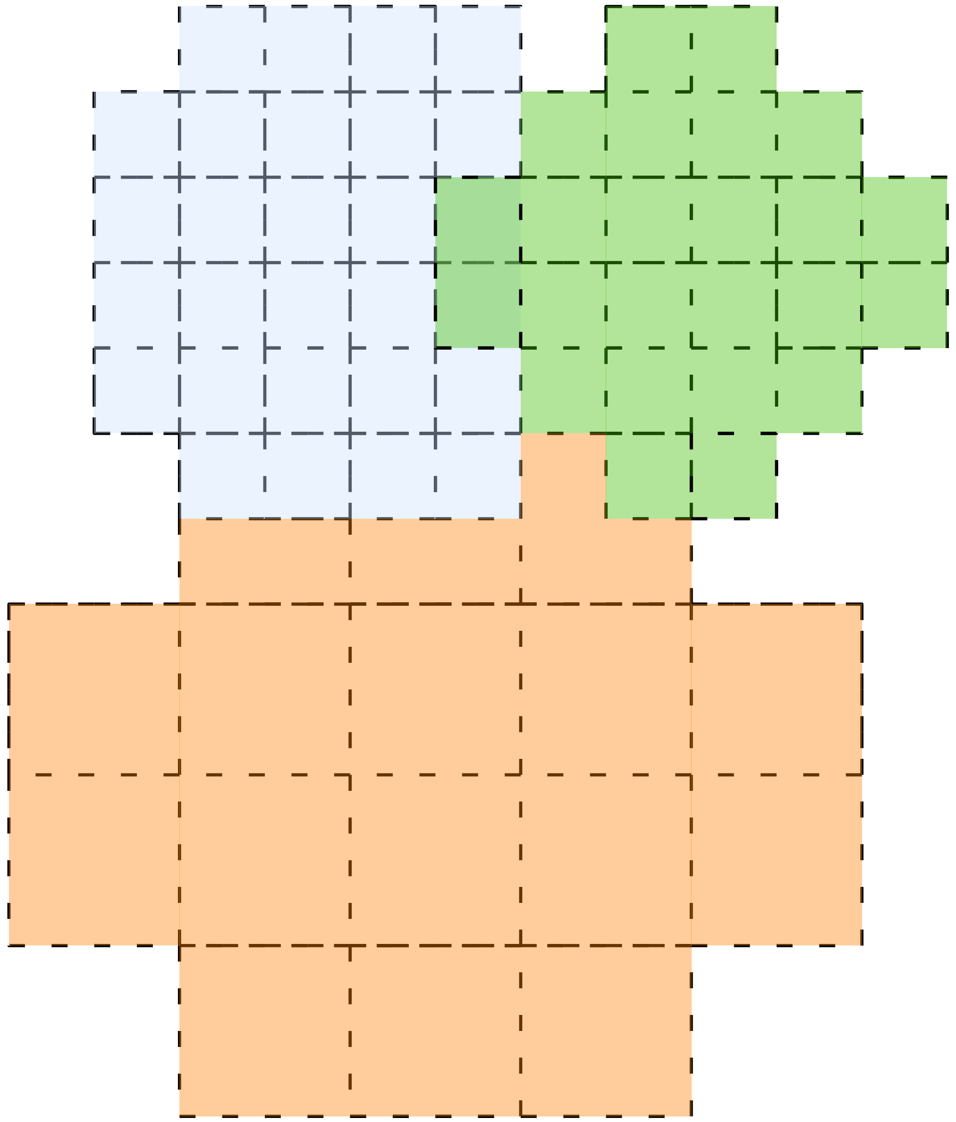
Creating the AVD

- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls
- Approximate b by the cells $\mathcal{C}'$ that intersect it
- Pick grid G_{2^i} s.t. $\sqrt{d}2^i \leq (\varepsilon/16)r$



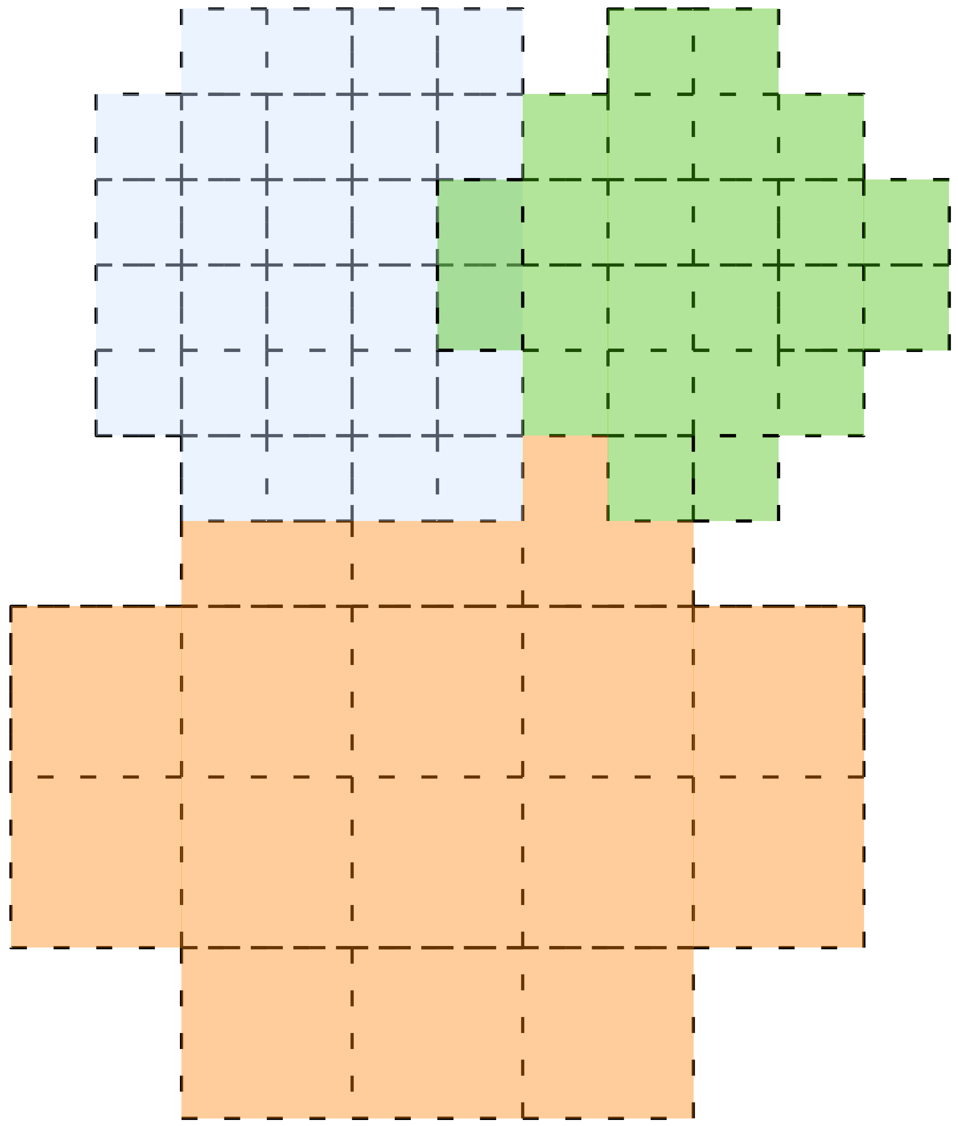
Creating the AVD

- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls
- Approximate b by the cells $\mathcal{C}'$ that intersect it
- Pick grid G_{2^i} s.t. $\sqrt{d}2^i \leq (\varepsilon/16)r$
- For each ball the amount of grid cells is bound by $O(\frac{1}{\varepsilon^d})$

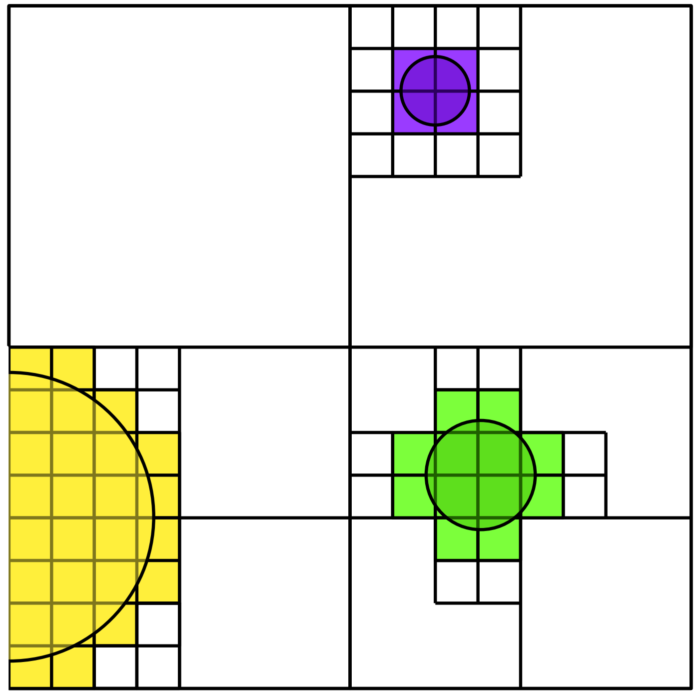


Creating the AVD

- Remember we can compute a set $\mathcal{B}$ of $O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$ balls
- Approximate b by the cells $\mathcal{C}'$ that intersect it
- Pick grid G_{2^i} s.t. $\sqrt{d}2^i \leq (\varepsilon/16)r$
- For each ball the amount of grid cells is bound by $O(\frac{1}{\varepsilon^d})$
- Create from $\mathcal{C}'$ a set $\mathcal{C}$ such that from each instance of $\square \in \mathcal{C}'$ we pick the $\square$ associated to the smallest ball

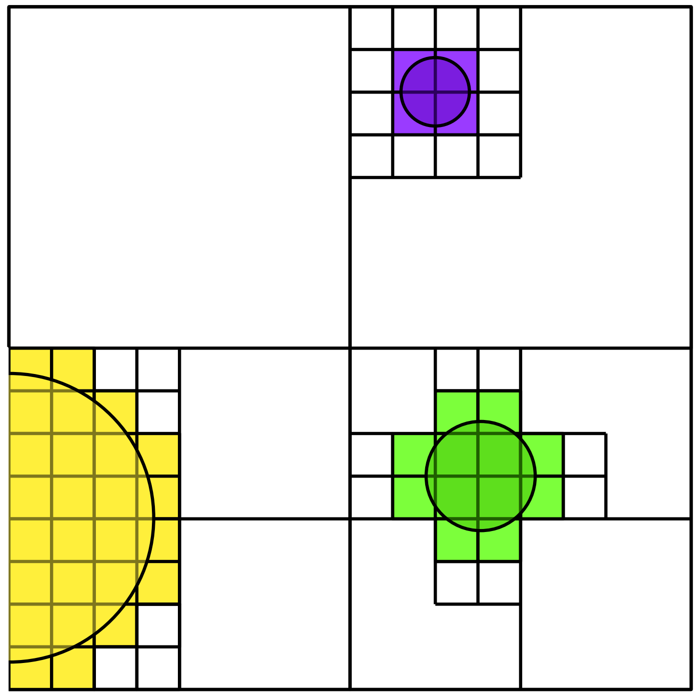


Point location on the grids



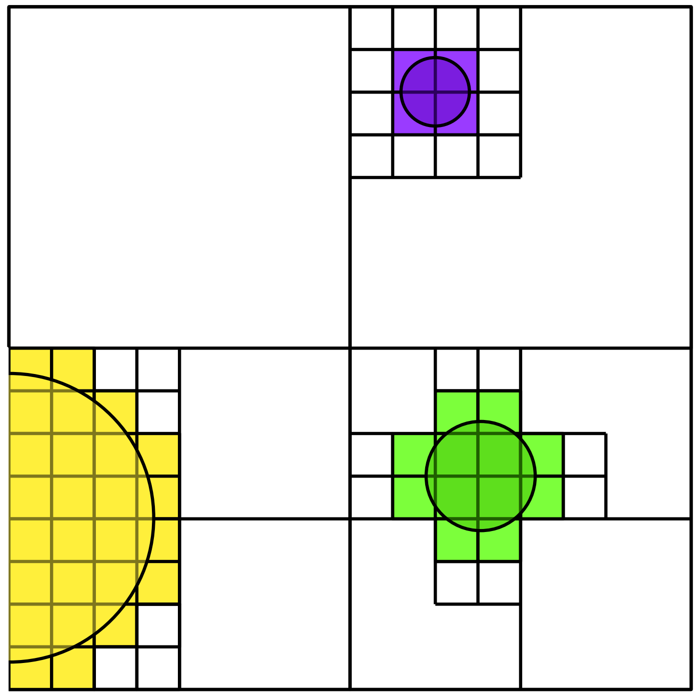
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$



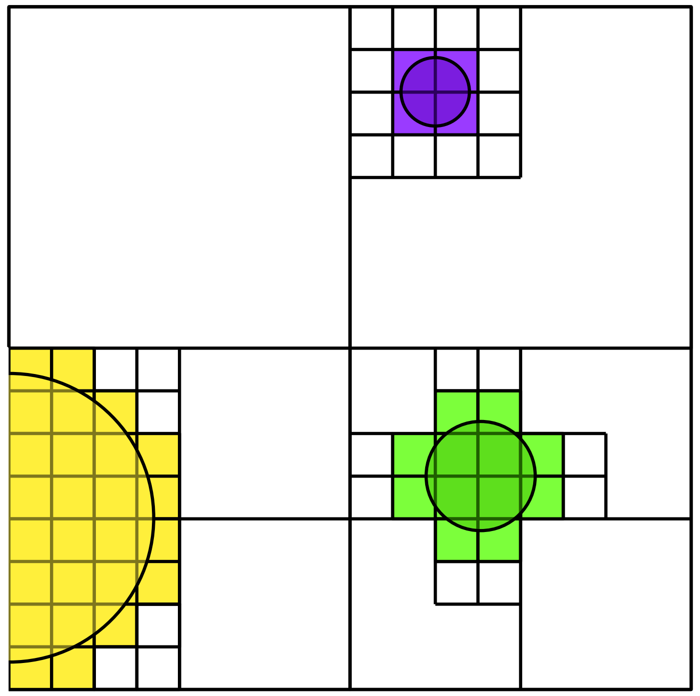
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$



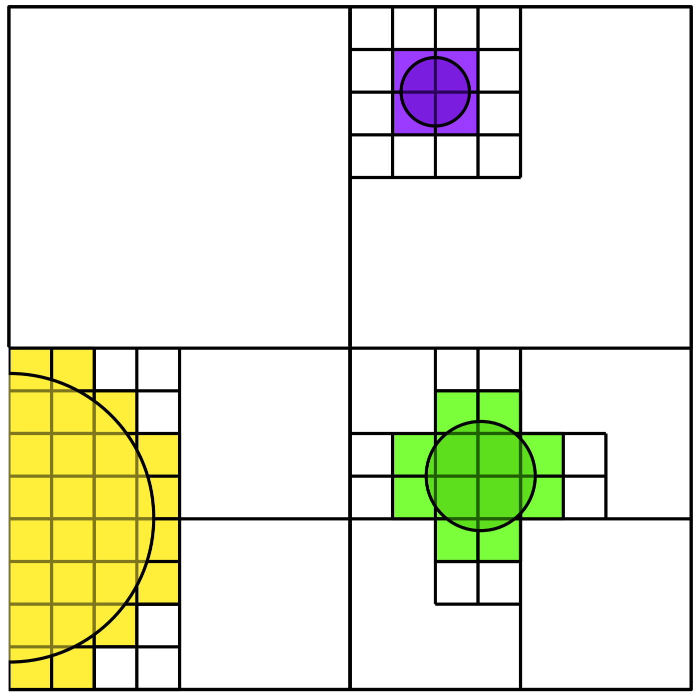
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$
- store cells in compressed quadtree!



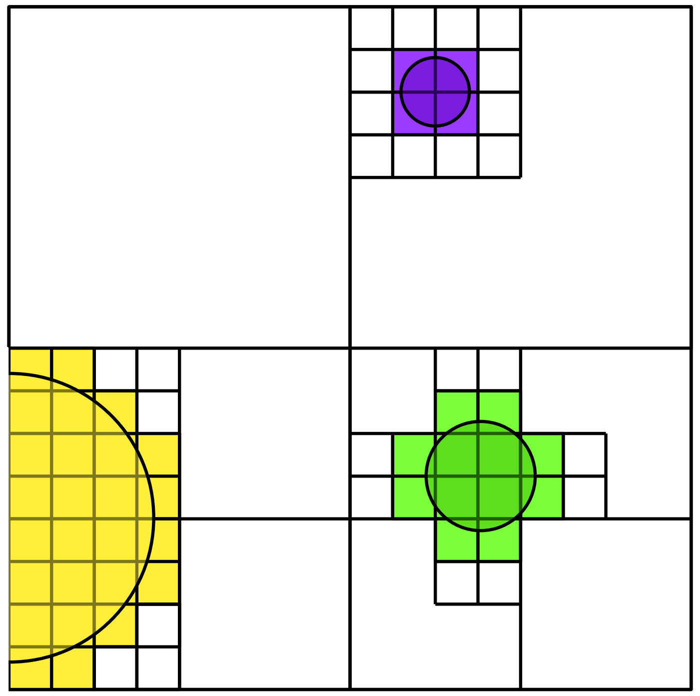
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$
- store cells in compressed quadtree!
- Construction: $O(|\mathcal{C}| \log |\mathcal{C}|)$ time



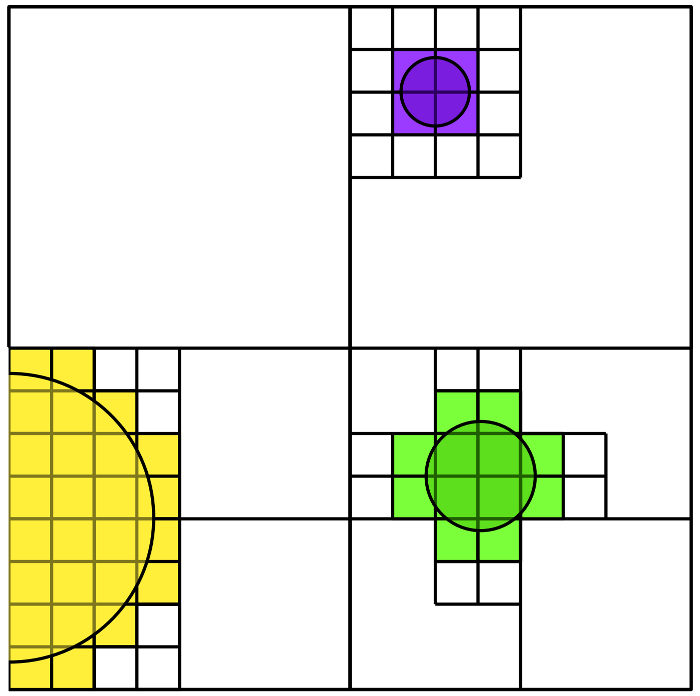
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$
- store cells in compressed quadtree!
- Construction: $O(|\mathcal{C}| \log |\mathcal{C}|)$ time
- Space: $O(|\mathcal{C}|)$



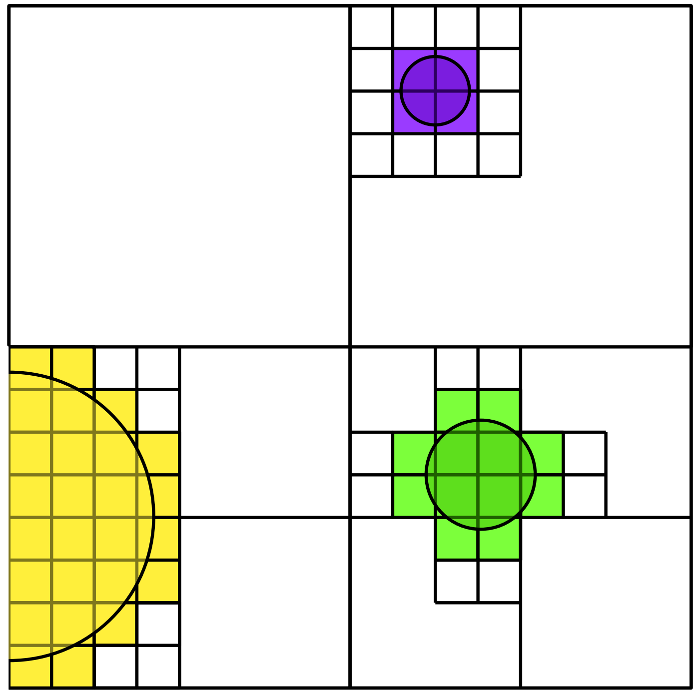
Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$
- store cells in compressed quadtree!
- Construction: $O(|\mathcal{C}| \log |\mathcal{C}|)$ time
- Space: $O(|\mathcal{C}|)$
- Query time: $O(\log |C|)$



Point location on the grids

- $(1+\varepsilon)$-ANN $\rightarrow$ target query on $\mathcal{B}_{\approx}$
- target query $\rightarrow$ find smallest canonical grid cell of $\mathcal{C}$
- store cells in compressed quadtree!
- Construction: $O(|\mathcal{C}| \log |\mathcal{C}|)$ time
- Space: $O(|\mathcal{C}|)$
- Query time: $O(\log |C|)$
- Store for each cell in a leaf the *smallest* ball it belongs to



Theorem:

Theorem:

Let P be a set of n points in $\mathbb{R}^d$. One can build a compressed quadtree $\hat{T}$ in:

Theorem:

Let P be a set of n points in $\mathbb{R}^d$. One can build a compressed quadtree $\hat{T}$ in:

- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$ time

Theorem:

Let P be a set of n points in $\mathbb{R}^d$. One can build a compressed quadtree $\hat{T}$ in:

- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$ time
- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$ size

Theorem:

Let P be a set of n points in $\mathbb{R}^d$. One can build a compressed quadtree $\hat{T}$ in:

- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$ time
- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$ size

Such that a $(1+\varepsilon)$-ANN query on P can be answered by a single point location query in $\hat{T}$ in:

- $O(\log \frac{n}{\varepsilon})$ time

Theorem:

Let P be a set of n points in $\mathbb{R}^d$. One can build a compressed quadtree $\hat{T}$ in:

- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$ time
- $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$ size

Such that a $(1+\varepsilon)$-ANN query on P can be answered by a single point location query in $\hat{T}$ in:

- $O(\log \frac{n}{\varepsilon})$ time

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon}$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}}$

$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$$

$$= \frac{1}{2d+2} \log \frac{n^{1/(2d+2)}}{\varepsilon}$$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$$= \frac{1}{2d+2} \log \frac{\boxed{n^{1/(2d+2)}}}{\varepsilon}$$

$$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$$

$$\boxed{n^{1/(2d+2)} \leq n}$$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$
- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$

$$= \frac{1}{2d+2} \log \frac{\boxed{n^{1/(2d+2)}}}{\varepsilon}$$

$$= O(\log \frac{n}{\varepsilon})$$

$\boxed{n^{1/(2d+2)} \leq n}$

Construction time: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

- Building a compressed quadtree can be done in $O(|C| \log |C|)$ time
- $|C|$ is naively bound by $N = O(\frac{|\mathcal{B}|}{\varepsilon^d})$
- $|C|$ can also be computed in that time
- $|\mathcal{B}| = O(\frac{n}{\varepsilon^{d+1}} \log \frac{1}{\varepsilon})$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

$$\boxed{\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})}$$

- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$$= \frac{1}{2d+2} \log \frac{\boxed{n^{1/(2d+2)}}}{\varepsilon} \longrightarrow \boxed{n^{1/(2d+2)} \leq n}$$

$$= O(\log \frac{n}{\varepsilon})$$

- $O(N \log N) = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$

Size: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

Size: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

- Compressed quadtrees have size $O(|C|)$

Size: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

- Compressed quadtrees have size $O(|C|)$
- $|C|$ is bound by $N = \frac{\mathcal{B}}{\varepsilon^d}$

Size: $O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

- Compressed quadtrees have size $O(|C|)$
- $|C|$ is bound by $N = \frac{\mathcal{B}}{\varepsilon^d}$
- $N = O(\frac{n}{\varepsilon^{2d+1}} \log \frac{1}{\varepsilon})$

Query time: $O(\log \frac{n}{\varepsilon})$

Query time: $O(\log \frac{n}{\varepsilon})$

- Compressed quadtrees query time $O(\log |C|)$

Query time: $O(\log \frac{n}{\varepsilon})$

- Compressed quadtrees query time $O(\log |C|)$
- $|C|$ is bound by $N = \frac{\mathcal{B}}{\varepsilon^d}$

Query time: $O(\log \frac{n}{\varepsilon})$

- Compressed quadtrees query time $O(\log |C|)$
- $|C|$ is bound by $N = \frac{\mathcal{B}}{\varepsilon^d}$
- $\log N = O(\log \frac{n}{\varepsilon})$

Query time: $O(\log \frac{n}{\varepsilon})$

- Compressed quadtrees query time $O(\log |C|)$
- $|C|$ is bound by $N = \frac{\mathcal{B}}{\varepsilon^d}$
- $\log N = O(\log \frac{n}{\varepsilon})$

$\log \frac{1}{\varepsilon} = O(\frac{1}{\varepsilon})$

- $\log N = \log \frac{n}{\varepsilon^{2d+1}} \boxed{\log \frac{1}{\varepsilon}} \leq \log \frac{n}{\varepsilon^{2d+2}}$

$$= \frac{1}{2d+2} \log \frac{\boxed{n^{1/(2d+2)}}}{\varepsilon} \longrightarrow n^{1/(2d+2)} \leq n$$

$$= O(\log \frac{n}{\varepsilon})$$

Summary

Summary

- Recap point-location among balls

Summary

- Recap point-location among balls
- Ball approximation

Summary

- Recap point-location among balls
- Ball approximation
- WSPD for size reduction

Summary

- Recap point-location among balls
- Ball approximation
- WSPD for size reduction
- Approximate Voronoi diagrams with proofs on the bounds