Approximate Nearest Neighbors

Low Dimensions

Approximate Nearest Neighbors

Low Dimensions

Overview

1. Introduction
2. ANN with quadtree (bounded spread)
3. Why low-quality approximation helps for unbounded spread
4. Low-quality approximation

Many Applications

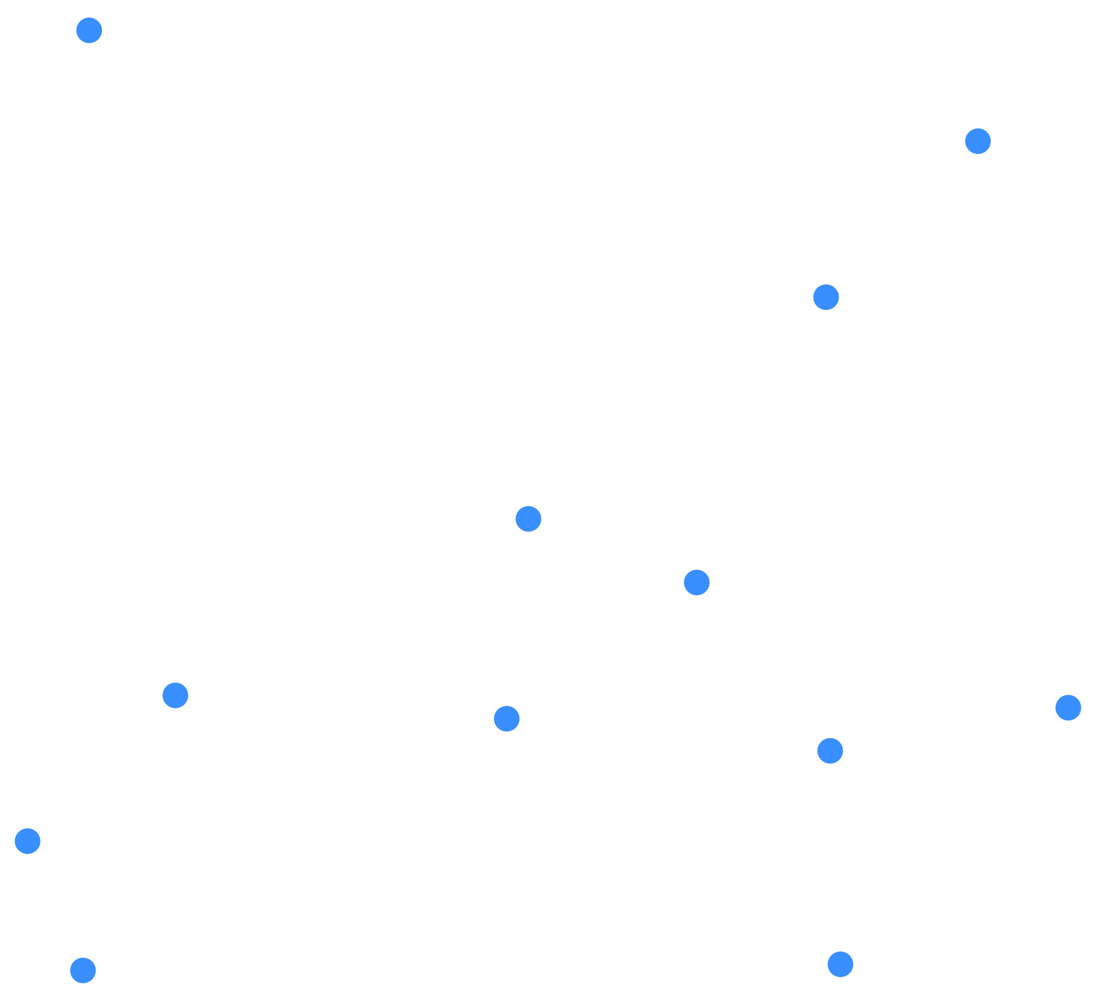
https://en.wikipedia.org/wiki/Nearest_neighbor_search

- Pattern recognition – in particular for optical character recognition
- Statistical classification – see k-nearest neighbor algorithm
- Computer vision
- Computational geometry – see Closest pair of points problem
- Databases – e.g. content-based image retrieval
- Coding theory – see maximum likelihood decoding
- Data compression – see MPEG-2 standard
- Robotic sensing[2]
- Recommendation systems, e.g. see Collaborative filtering
- Internet marketing – see contextual advertising and behavioral targeting
- DNA sequencing
- Spell checking – suggesting correct spelling
- Plagiarism detection
- Similarity scores for predicting career paths of professional athletes.
- Cluster analysis – assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense, usually based on Euclidean distance
- Chemical similarity
- Sampling-based motion planning

Exact nearest neighbor

Problem statement

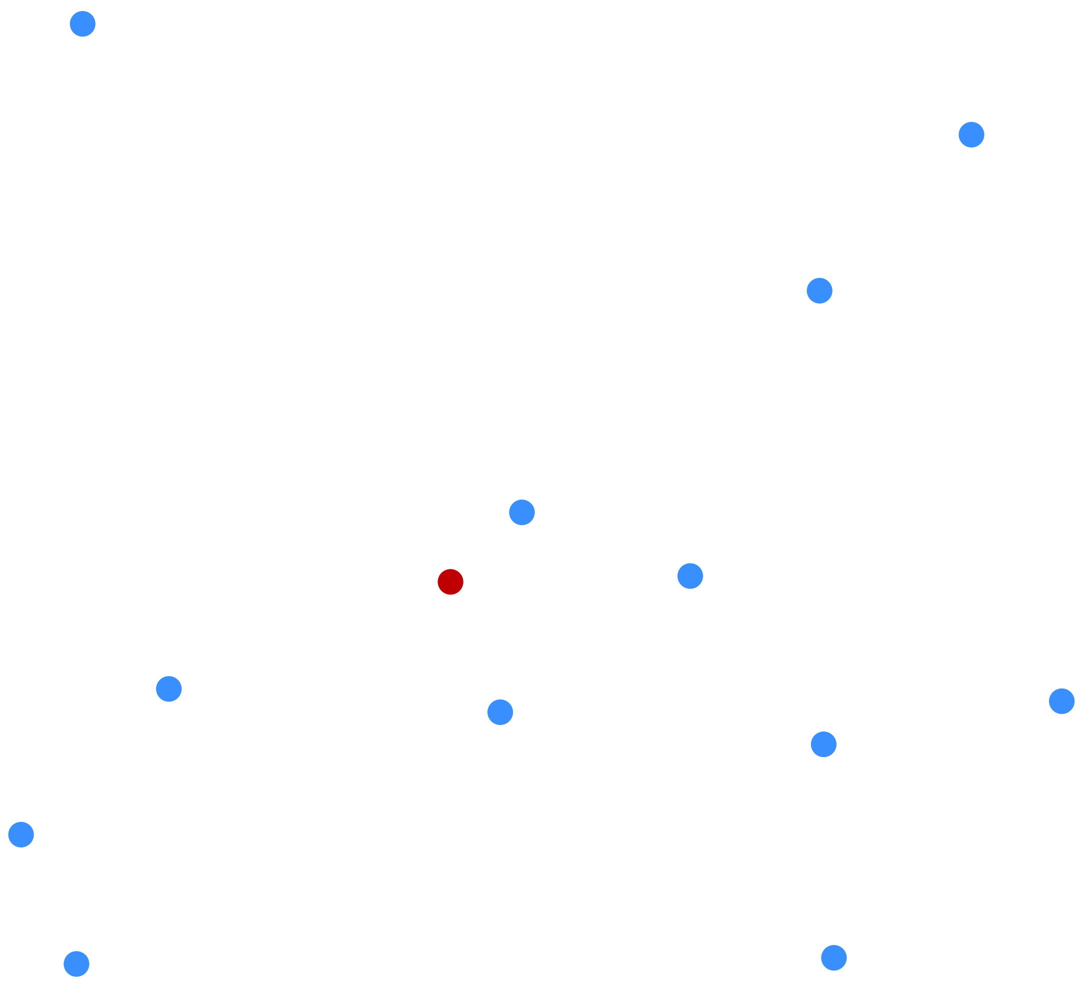
Preprocess set P of n points in $\mathbb{R}^d$ such that
given a query point q, we can find the closest point in P to q quickly.



Exact nearest neighbor

Problem statement

Preprocess set P of n points in $\mathbb{R}^d$ such that
given a query point q, we can find the closest point in P to q quickly.

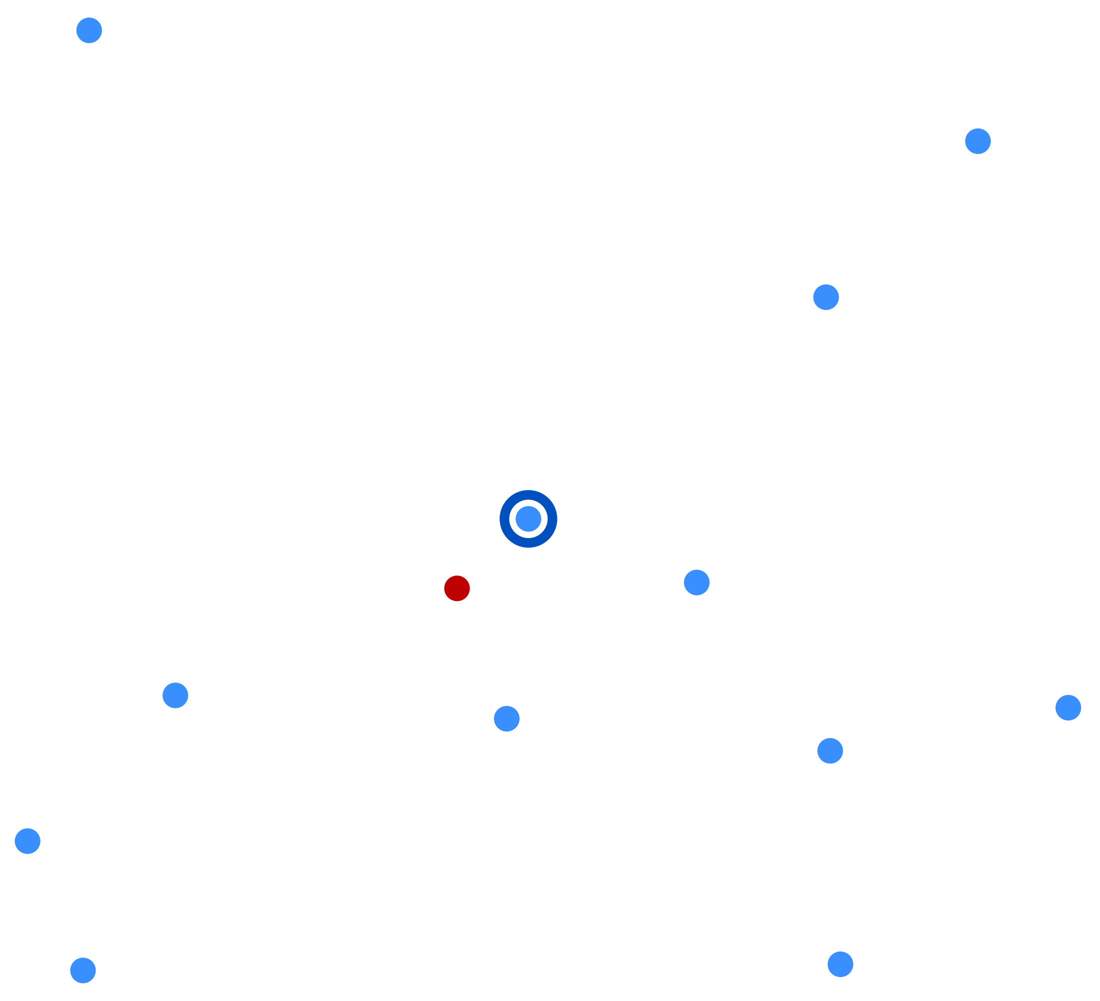


Exact nearest neighbor

Problem statement

Preprocess set P of n points in $\mathbb{R}^d$ such that

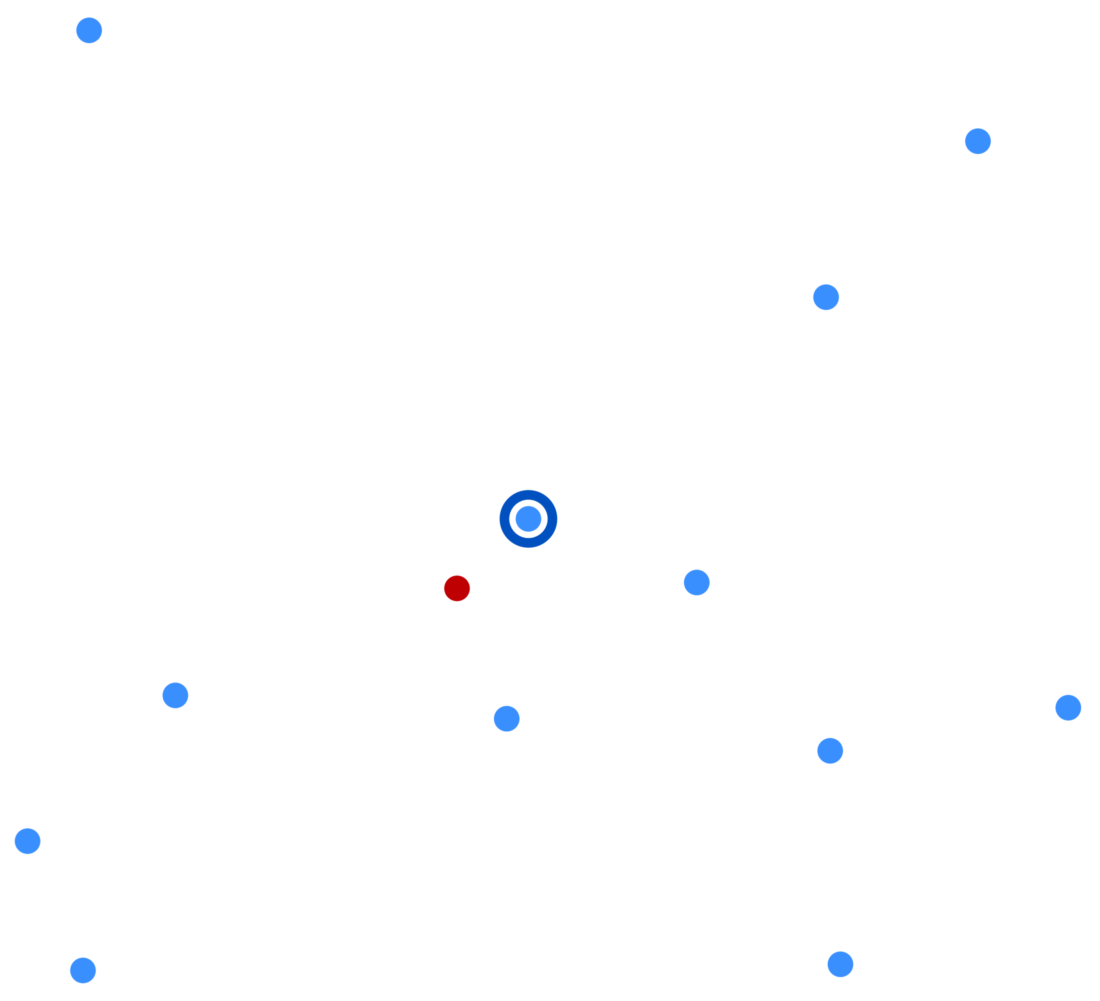
given a query point q, we can find the closest point in P to q quickly.



Exact nearest neighbor

Problem statement

Preprocess set P of n points in $\mathbb{R}^d$ such that
given a query point q, we can find the closest point in P to q quickly.

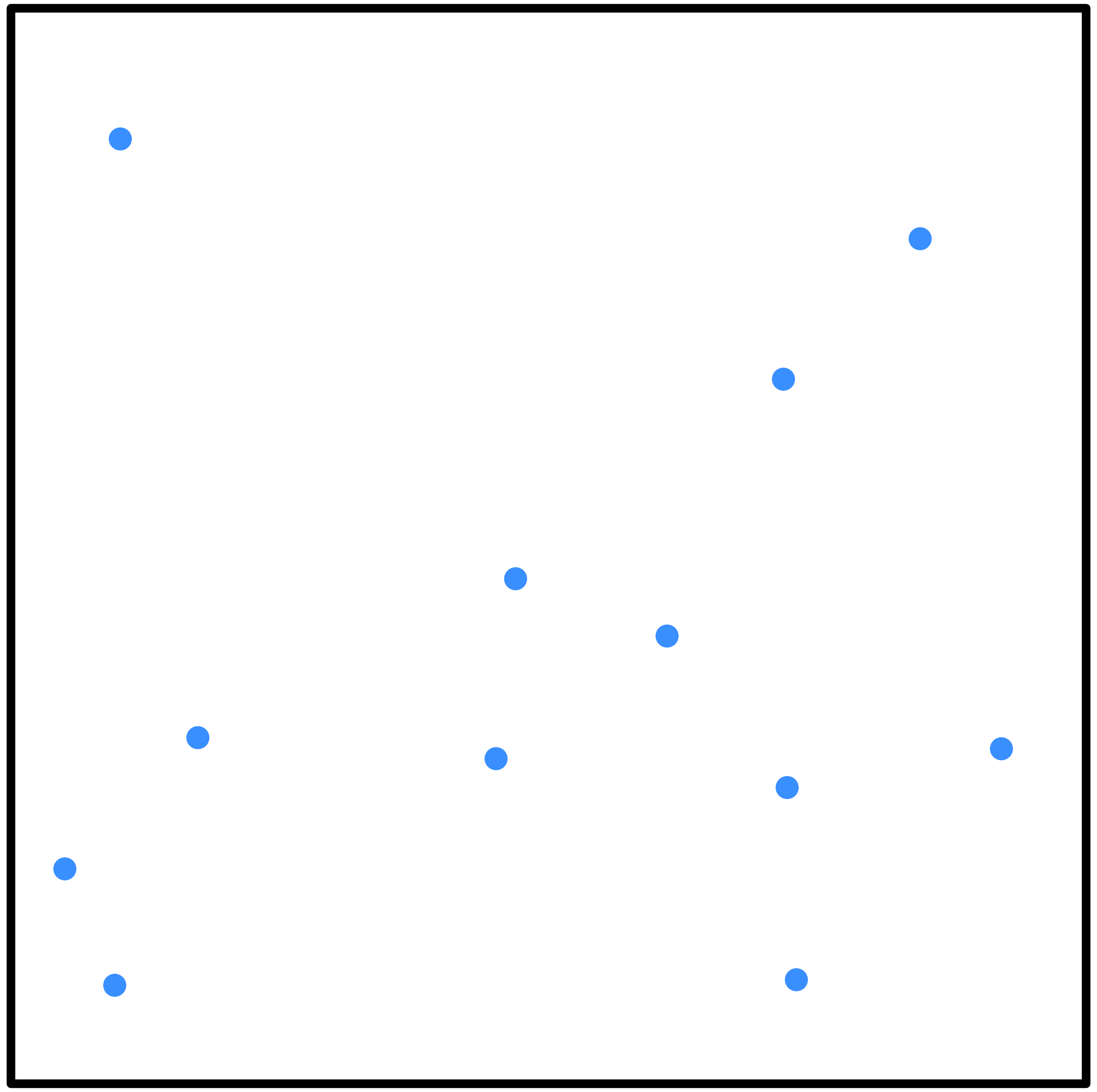


notation:

Nearest neighbor of q: $nn(q) = nn(q, P)$

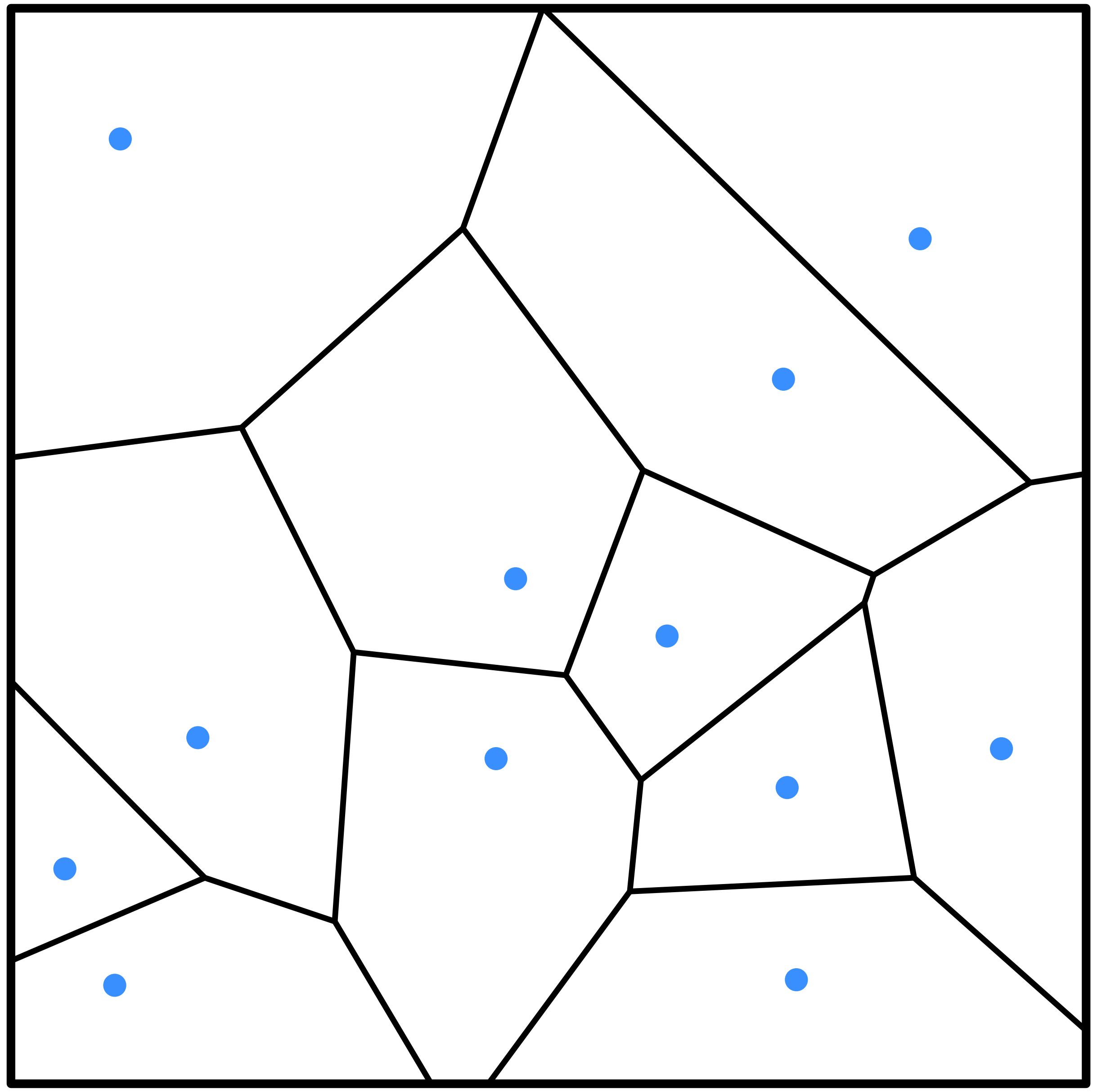
$d(q, P) = \|q - nn(q)\|$

Exact nearest neighbor



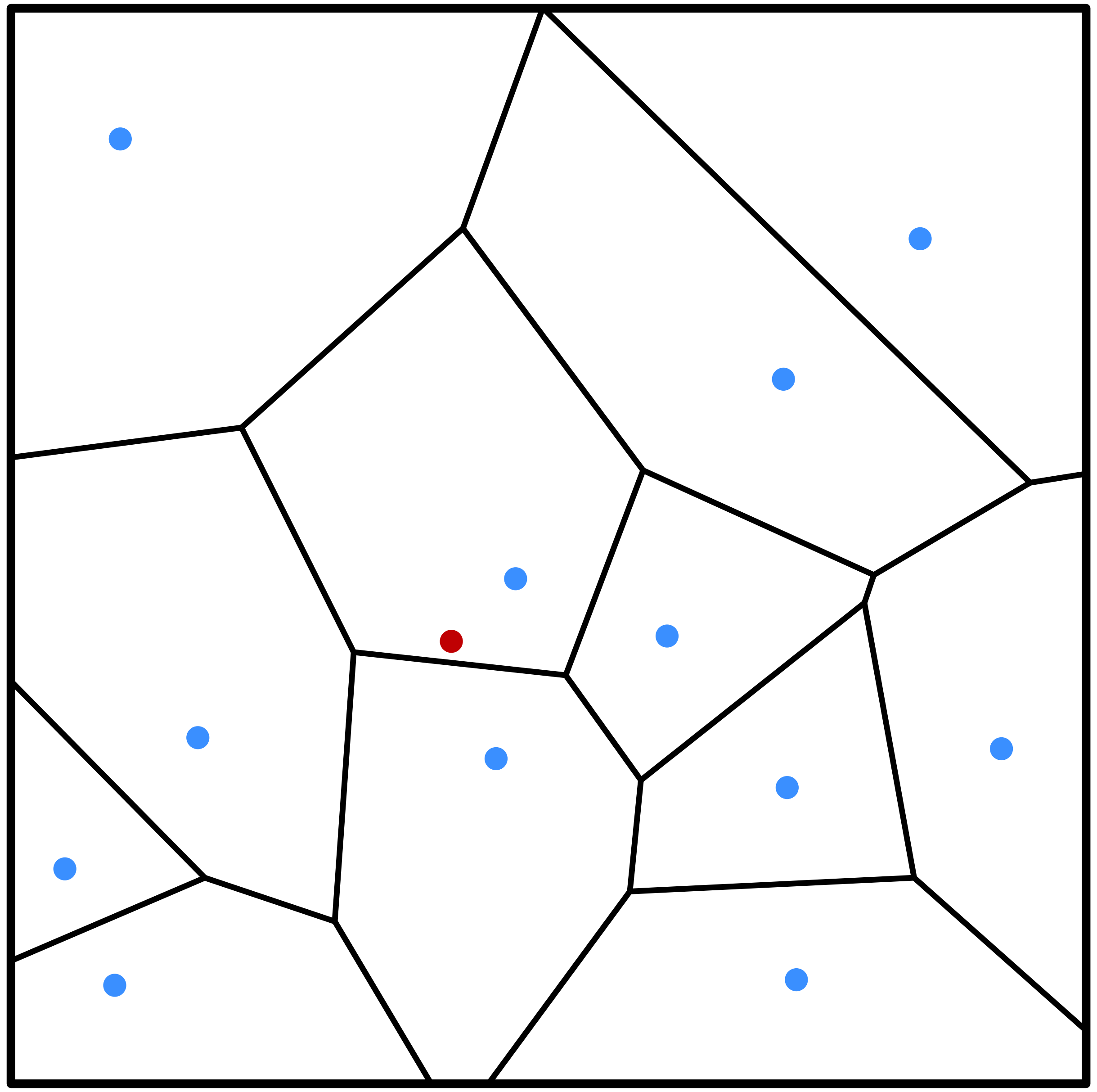
Exact nearest neighbor

Voronoi diagram



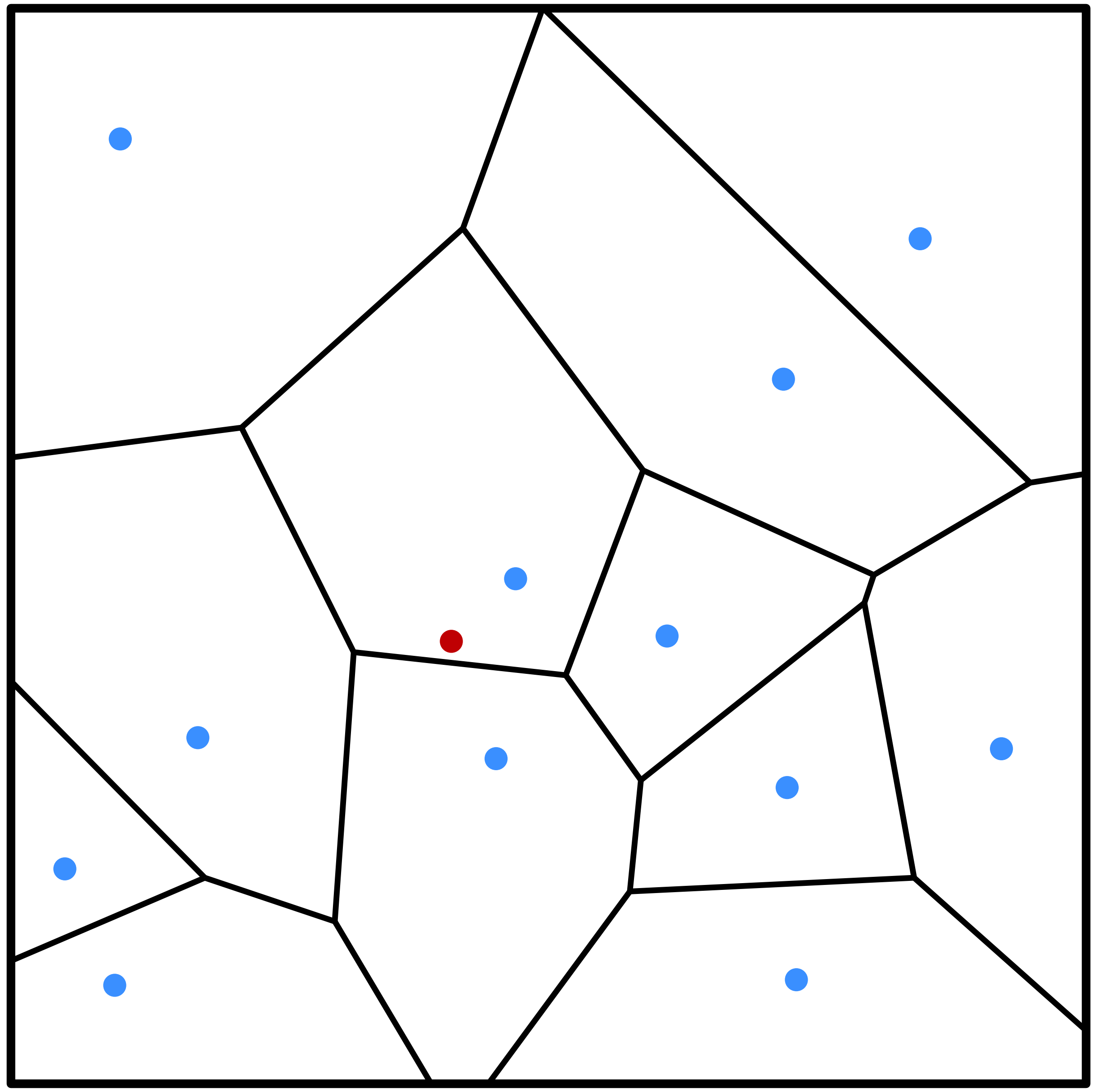
Exact nearest neighbor

Voronoi diagram



Exact nearest neighbor

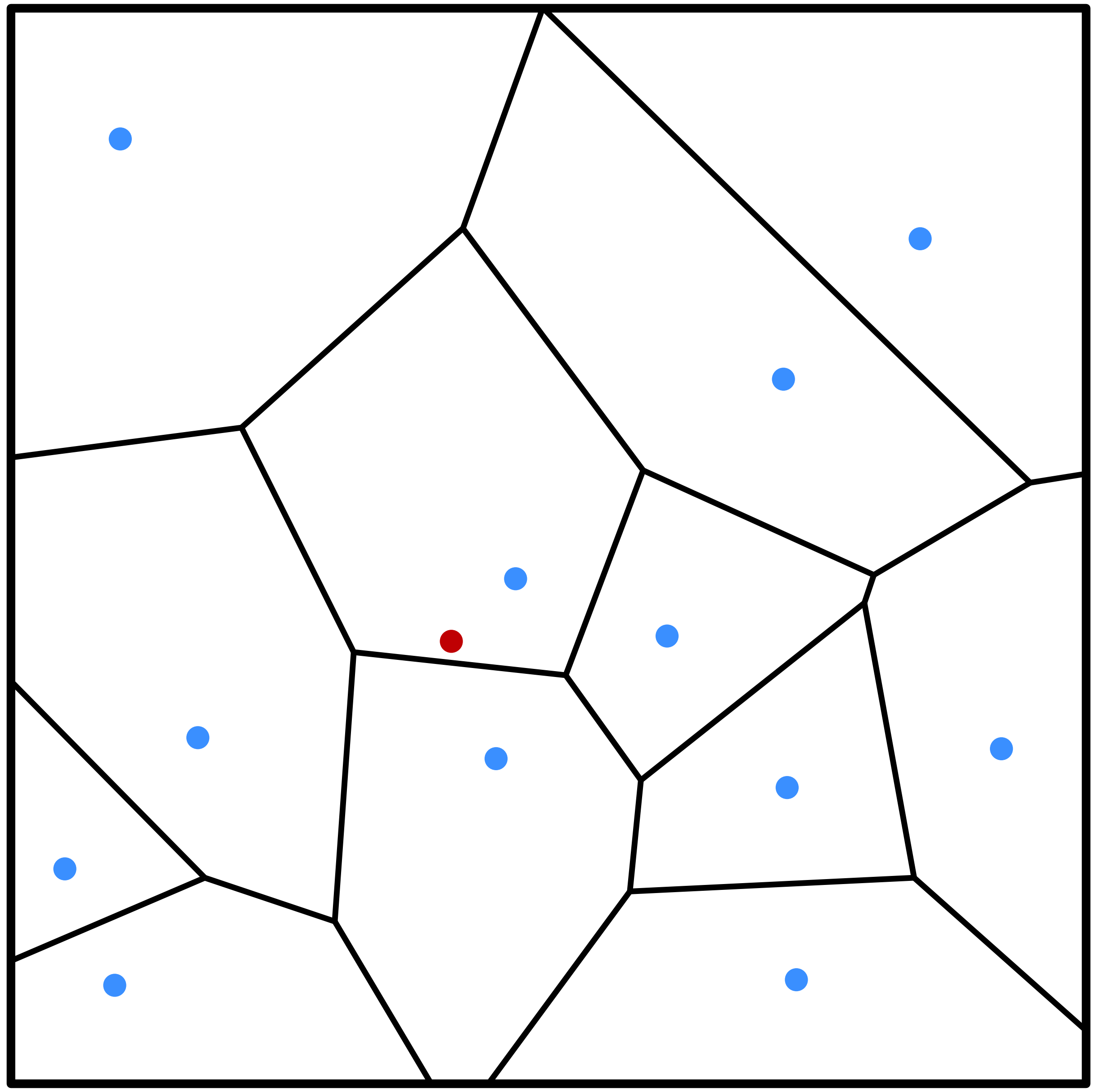
Voronoi diagram



Computing the Voronoi diagram of P and preprocessing it for point-location queries requires roughly $O(n^{\lceil d/2 \rceil} + n \log n)$ time.

Exact nearest neighbor

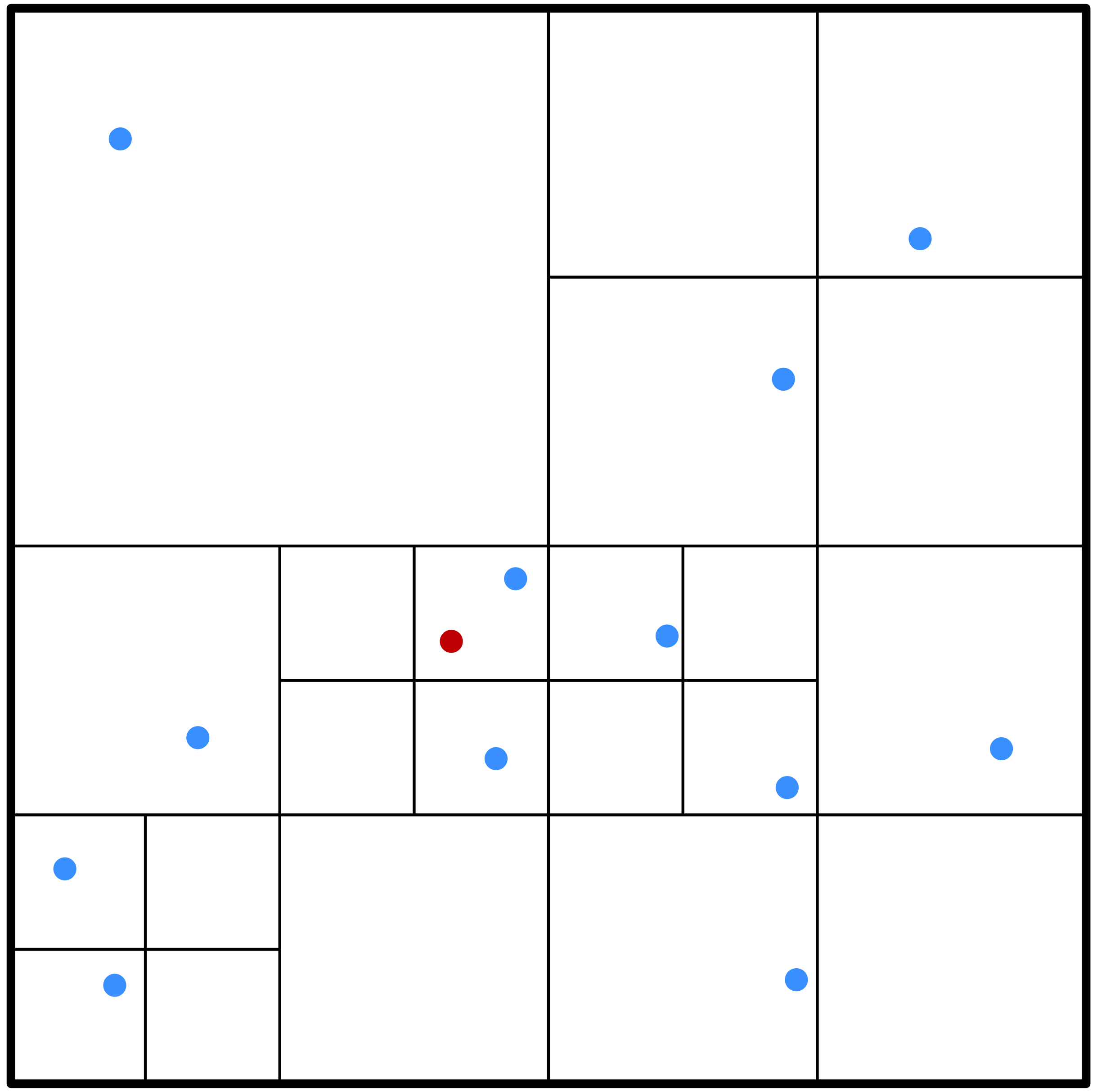
Voronoi diagram



Computing the Voronoi diagram of P and preprocessing it for point-location queries requires roughly $O(n^{\lceil d/2 \rceil} + n \log n)$ time.

Faster approximation?

Exact nearest neighbor



Computing the Voronoi diagram of P and preprocessing it for point-location queries requires roughly $O(n^{\lceil d/2 \rceil} + n \log n)$ time.

Faster approximation?

How about quadtrees?

Approximate nearest neighbor (Bounded spread)

Approximate nearest neighbor (Bounded spread)

$s \in P$ is a $(1+\varepsilon)$-approximate nearest neighbor (ANN) of q
if $||q - s|| \leq (1+\varepsilon)d(q, P)$.

Approximate nearest neighbor (Bounded spread)

$s \in P$ is a $(1+\varepsilon)$-approximate nearest neighbor (ANN) of q
if $||q-s|| \leq (1+\varepsilon)d(q,P)$.

Spread: $\Phi(P) = \frac{\max_{p,q \in P} ||p-q||}{\min_{p,q \in P, p \neq q} ||p-q||}$

Approximate nearest neighbor (Bounded spread)

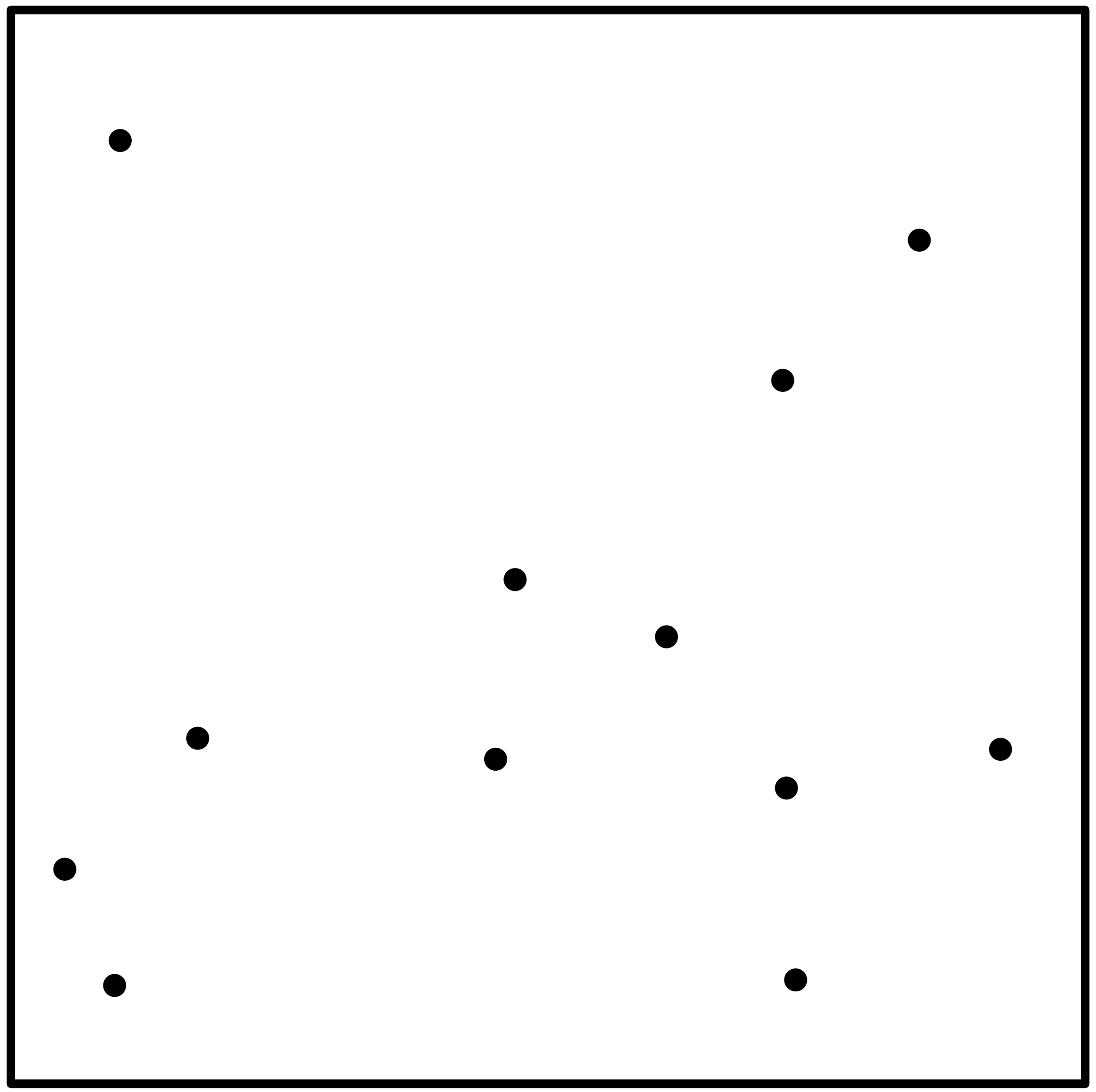
$s \in P$ is a $(1+\varepsilon)$-approximate nearest neighbor (ANN) of q
if $||q - s|| \leq (1+\varepsilon)d(q, P)$.

Spread: $\Phi(P) = \frac{\max_{p,q \in P} ||p-q||}{\min_{p,q \in P, p \neq q} ||p-q||}$

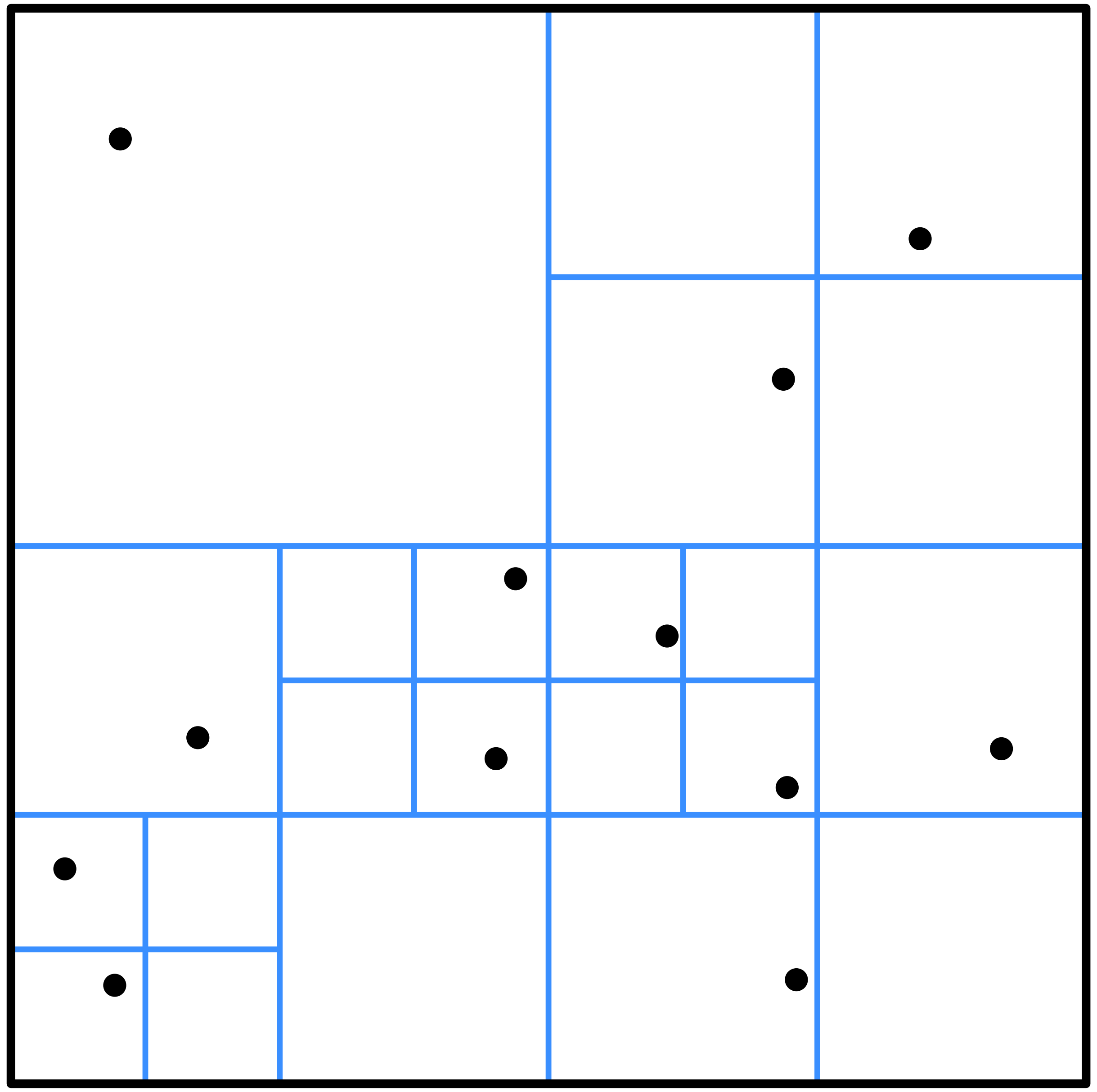
Setting:

- $P \subset [0,1]^d$, diameter$(P) = \Omega(1)$, $\Phi(P) = O(n^c)$, for constant c.
- $\mathcal{T}$: quadtree of P
- $\text{rep}_u \in P$: representative of node $u \in \mathcal{T}$
- $\varepsilon > 0$
- query point q

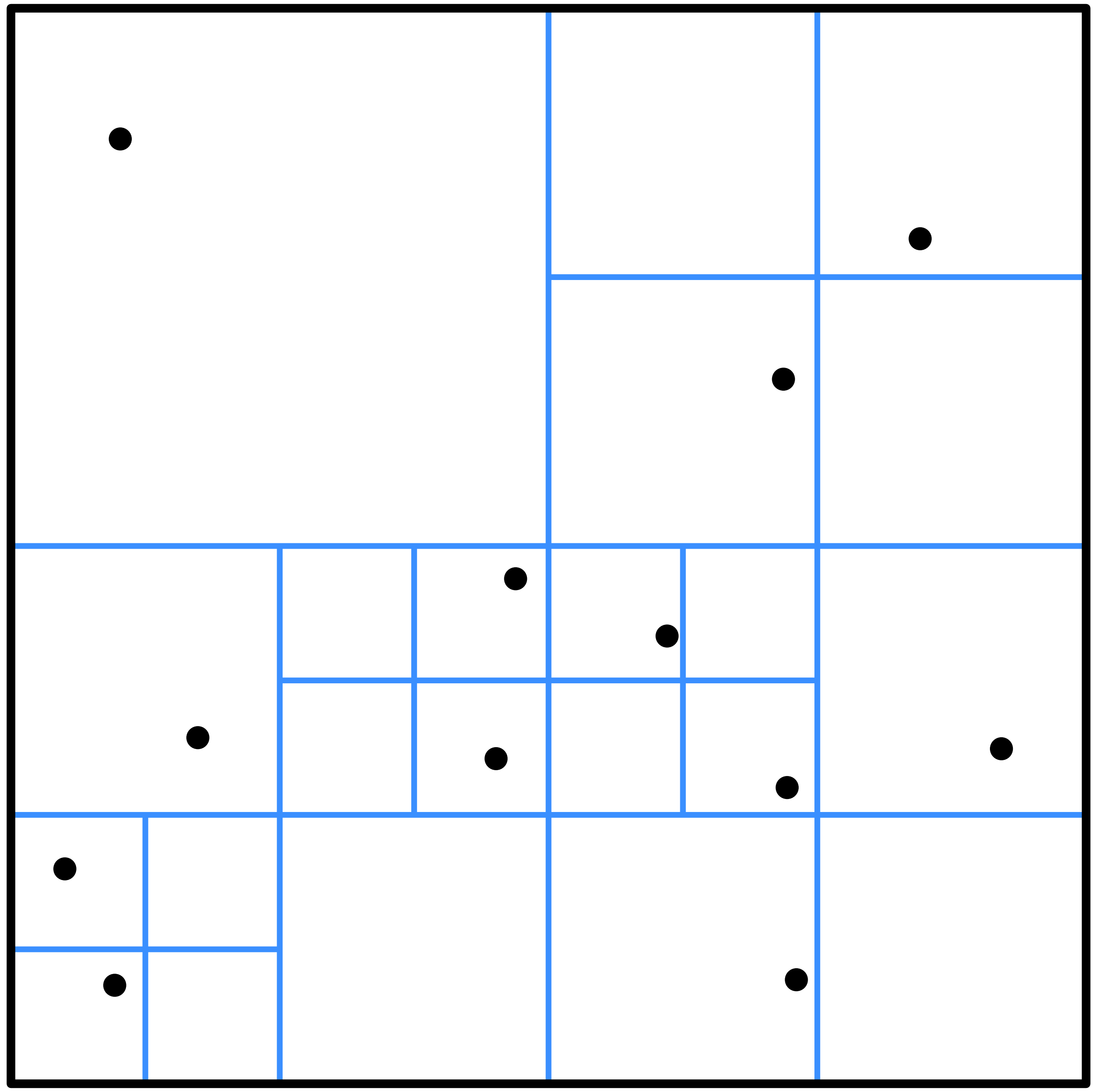
Approximate nearest neighbor (Bounded spread)

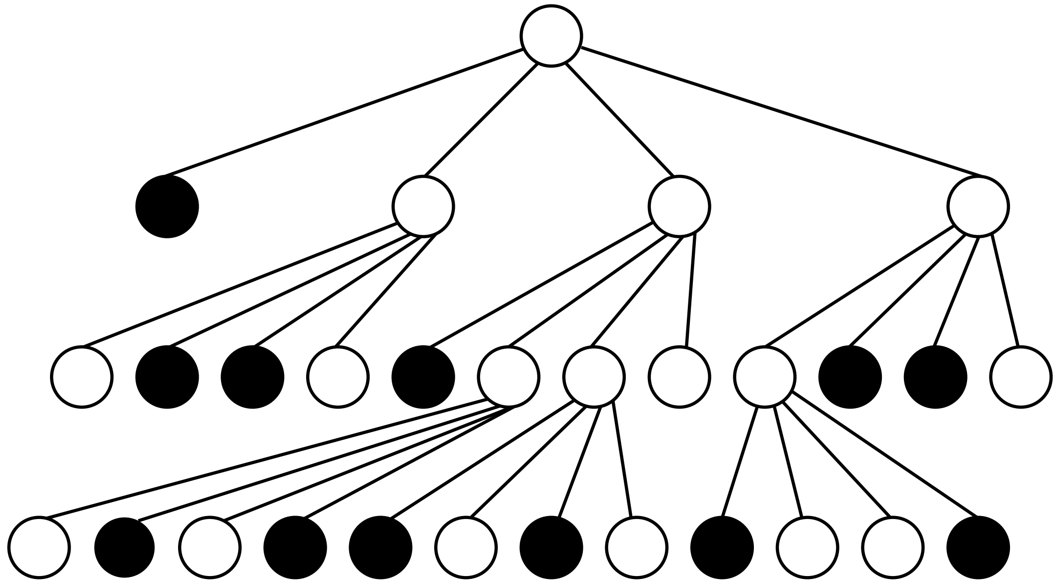


Approximate nearest neighbor (Bounded spread)

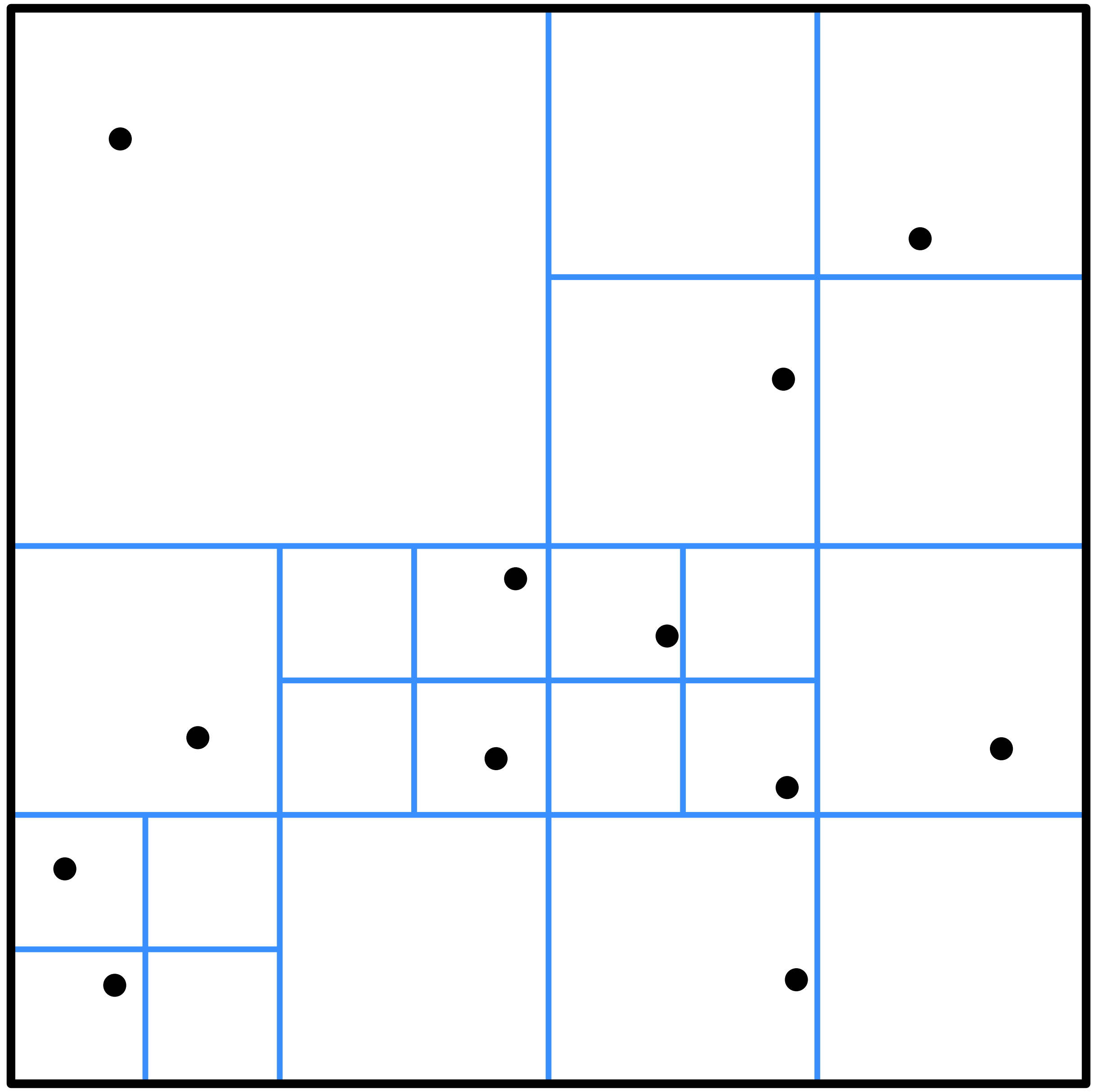


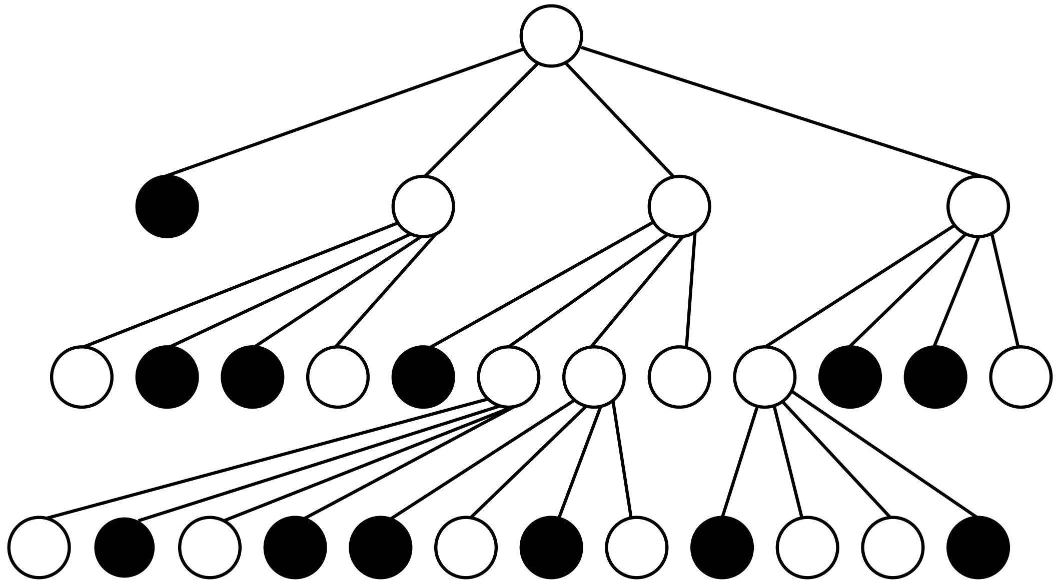
Approximate nearest neighbor (Bounded spread)





Approximate nearest neighbor (Bounded spread)

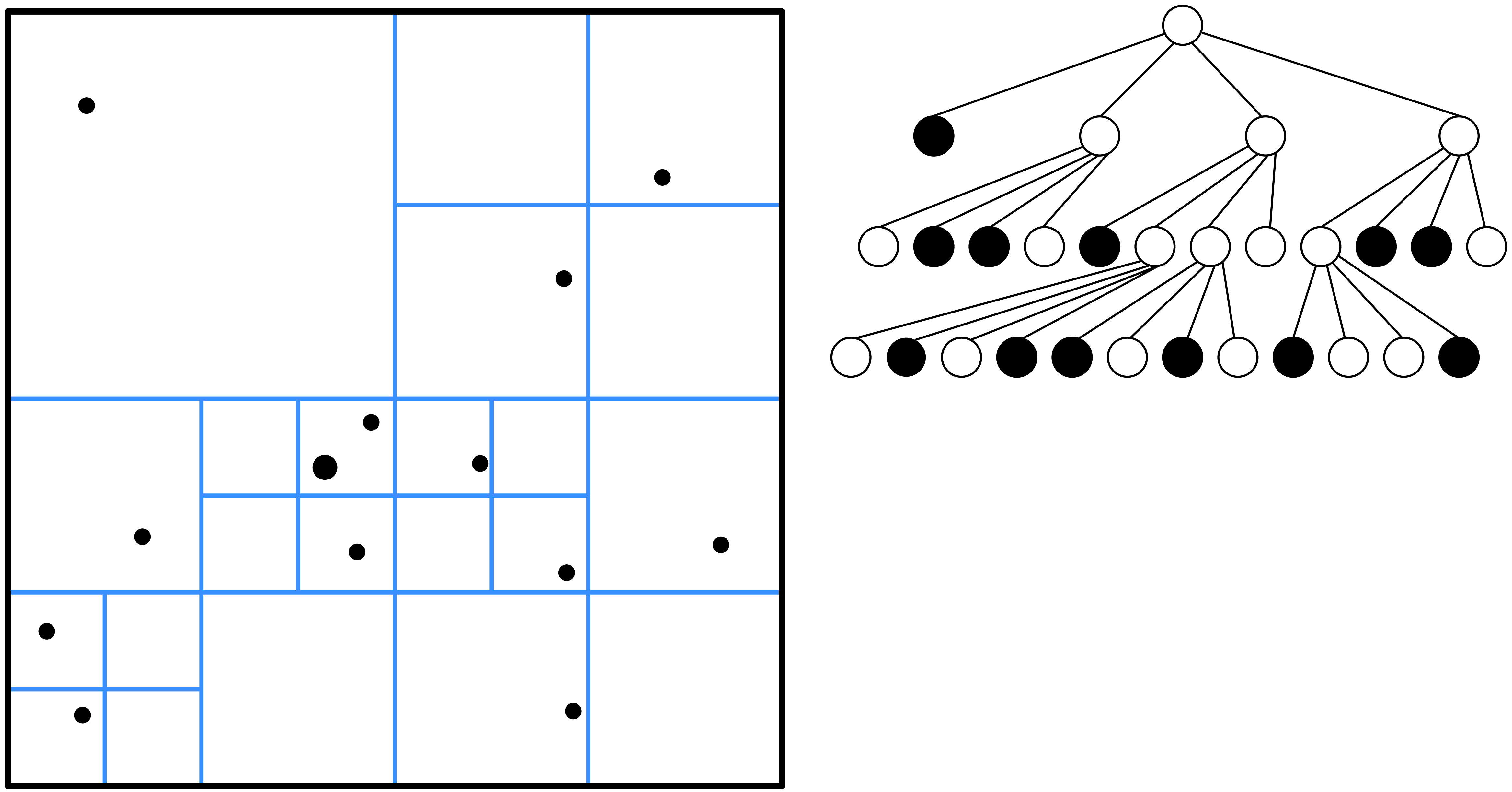




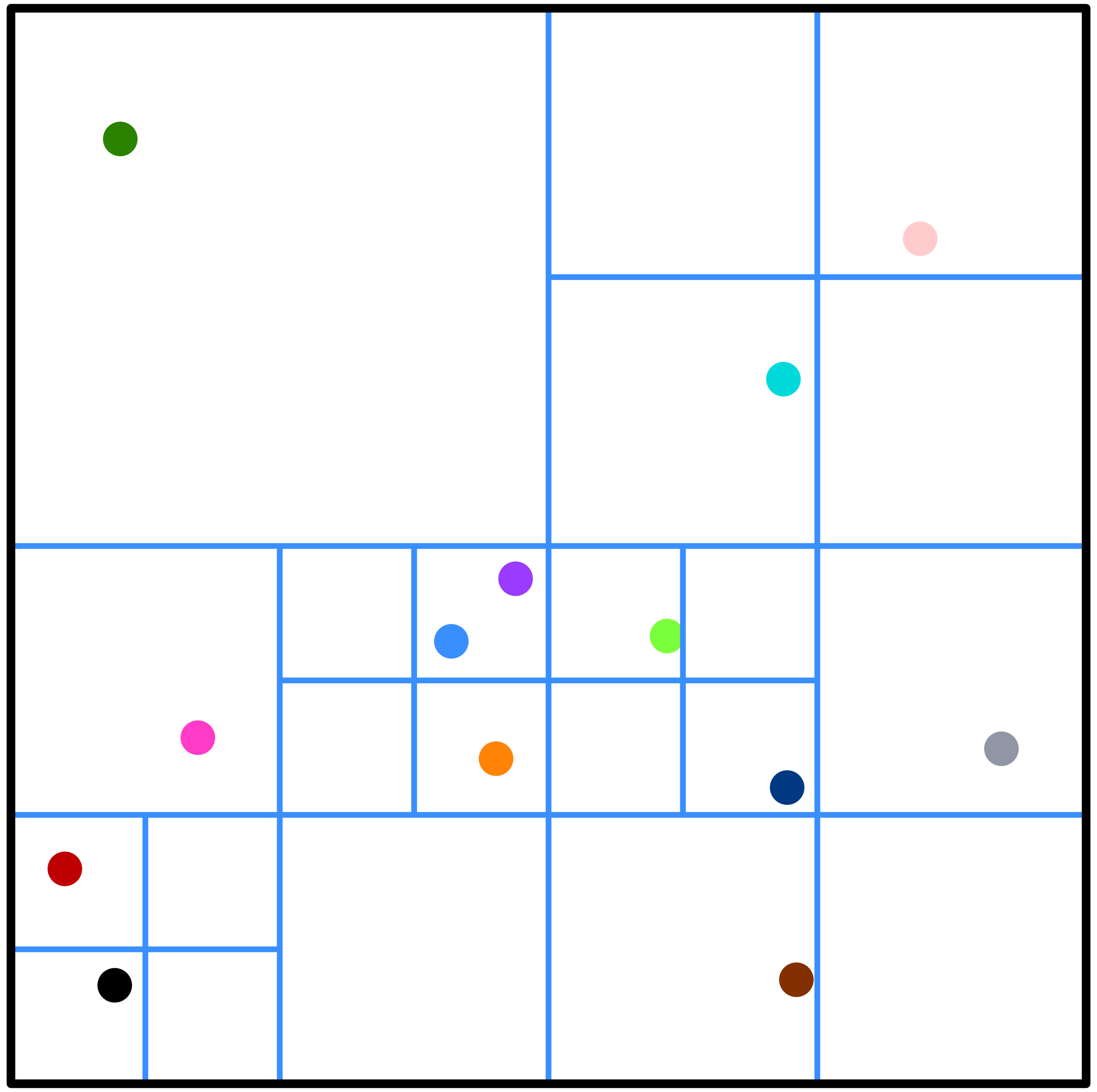
Questions:

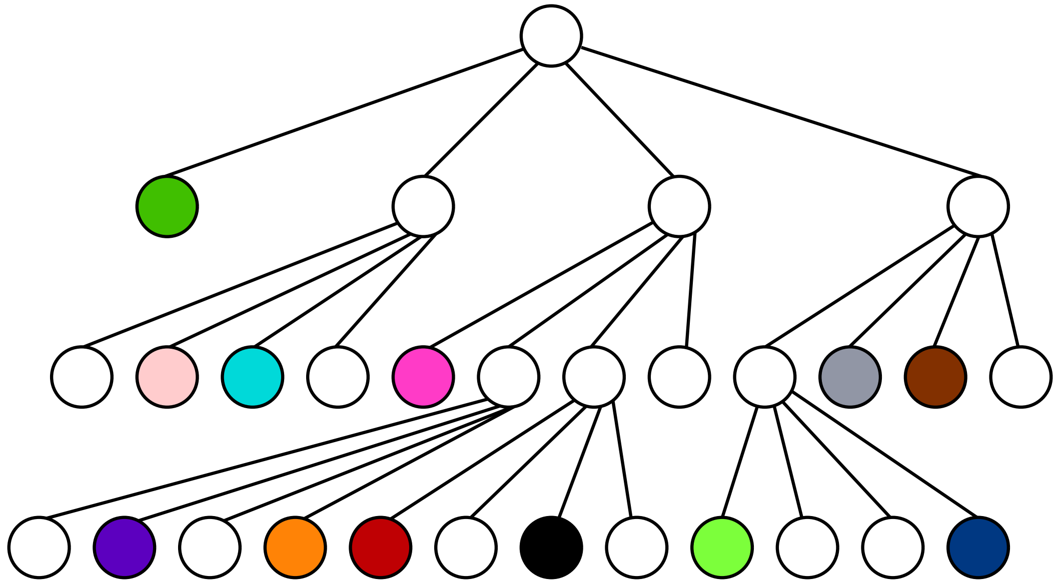
How long does point location take in a quadtree? How long in a compressed quadtree?

Approximate nearest neighbor (Bounded spread)

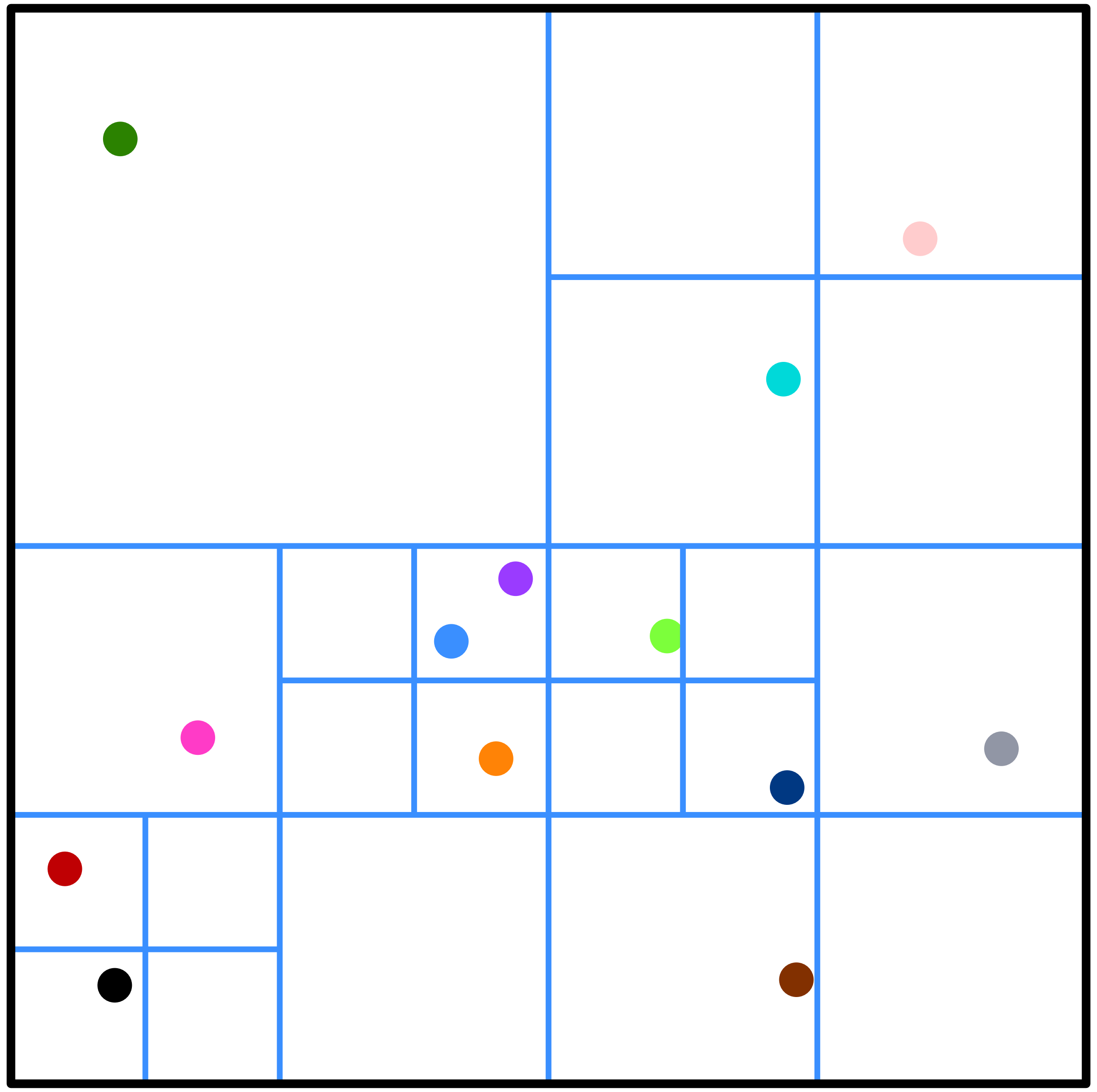


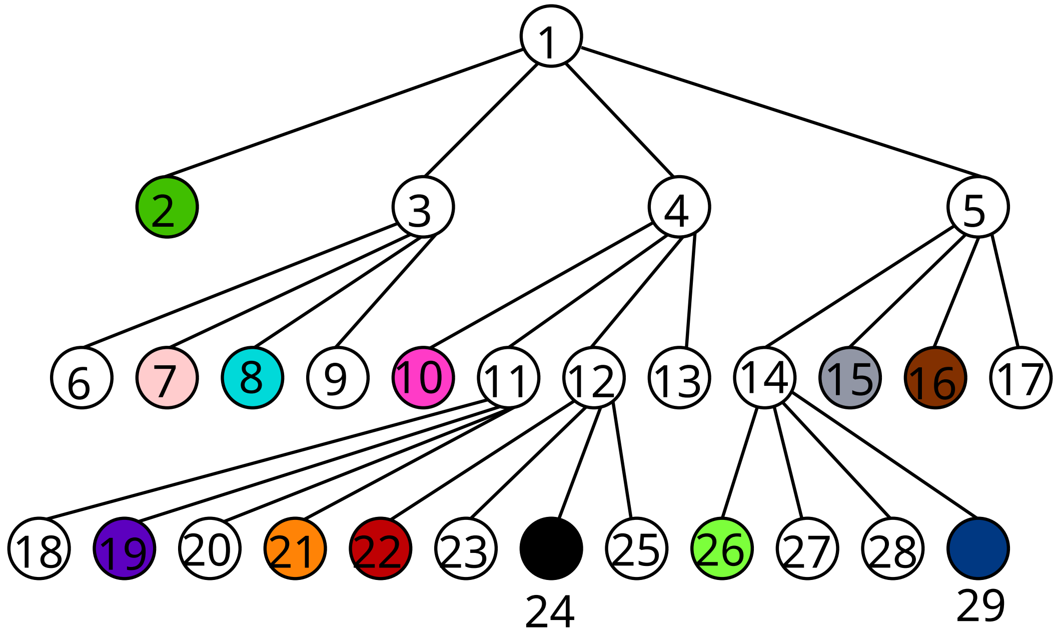
Approximate nearest neighbor (Bounded spread)





Approximate nearest neighbor (Bounded spread)





Ideas for ANN?

Approximate nearest neighbor (Bounded spread)

Algorithm ideas

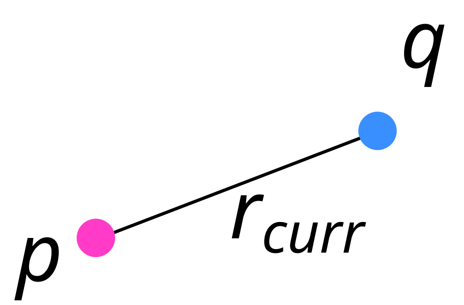
recursive: start at root (like point location).

Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$



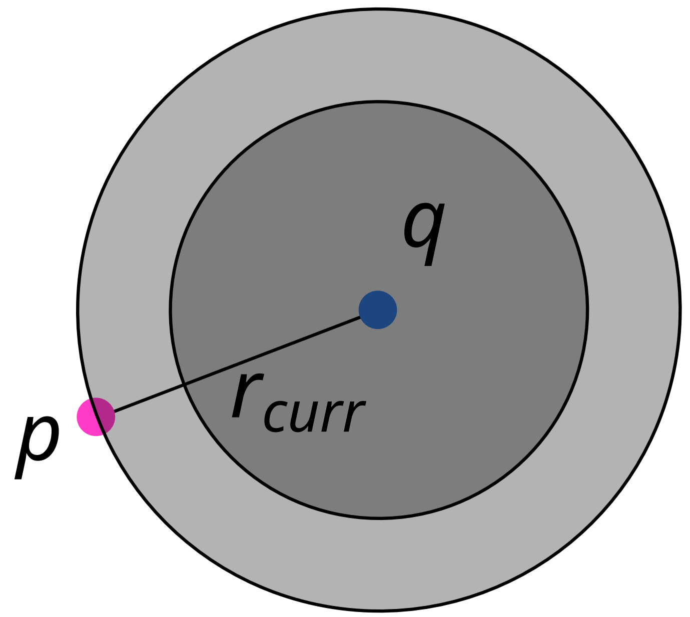
Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

only recurse on nodes/squares that could decrease distance significantly



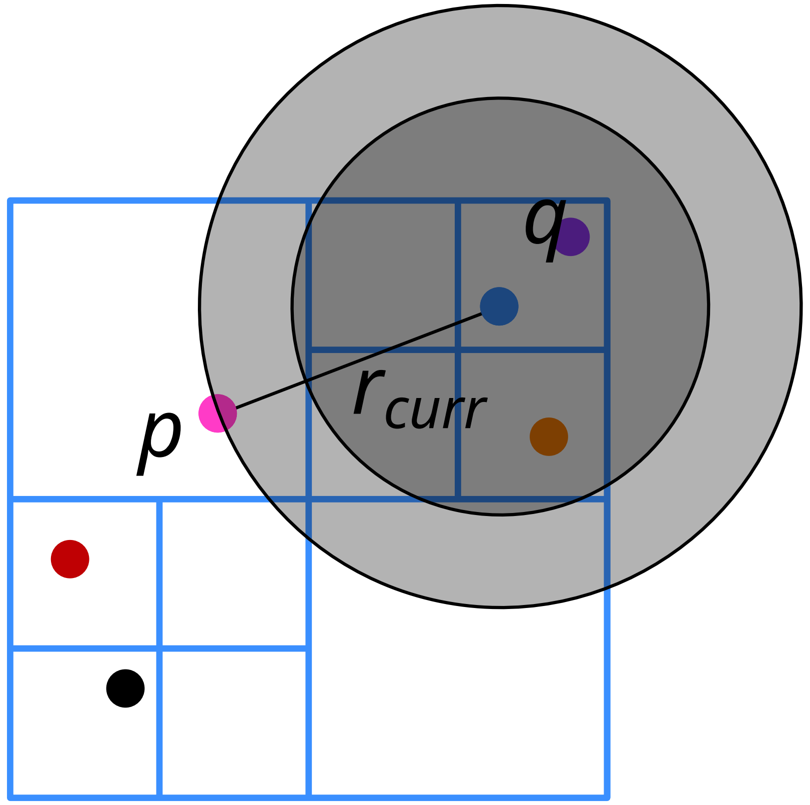
Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

only recurse on nodes/squares that could decrease distance significantly



could contain $s \in P$ with $||q - s|| < (1 - \varepsilon/2)r_{curr}$

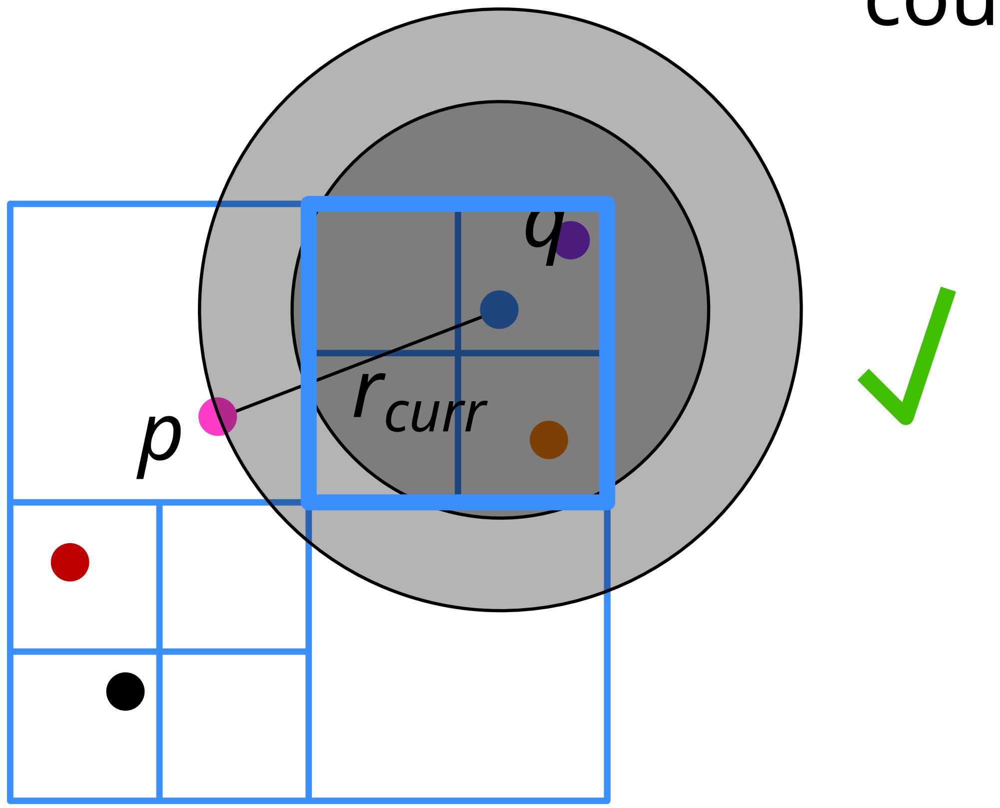
Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

only recurse on nodes/squares that could decrease distance significantly



could contain $s \in P$ with $||q - s|| < (1 - \varepsilon/2)r_{curr}$

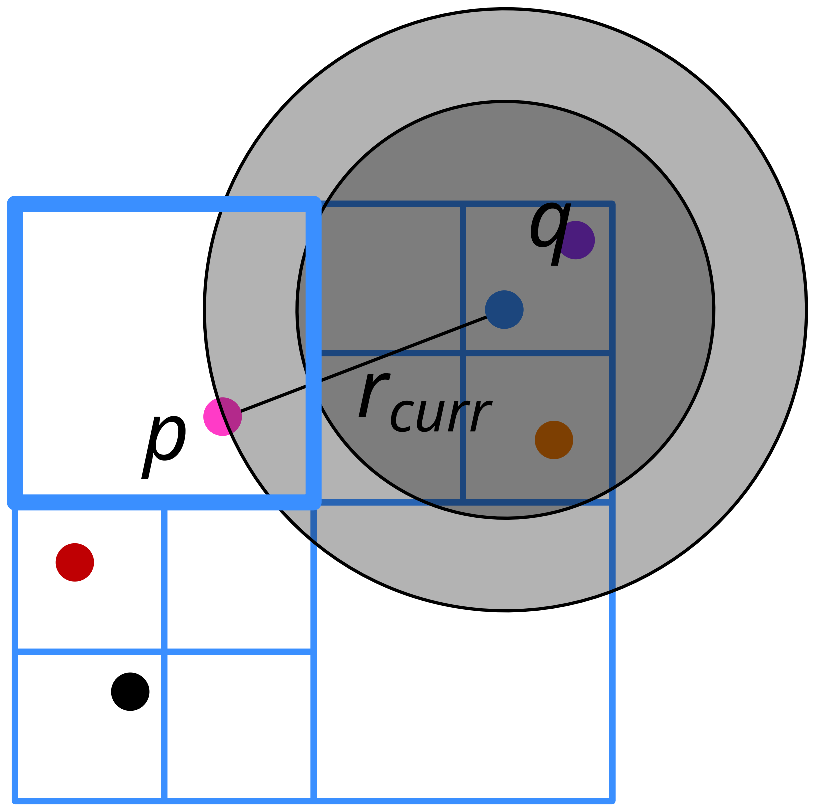
Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

only recurse on nodes/squares that could decrease distance significantly



could contain $s \in P$ with $||q - s|| < (1 - \varepsilon/2)r_{curr}$

✓

Approximate nearest neighbor (Bounded spread)

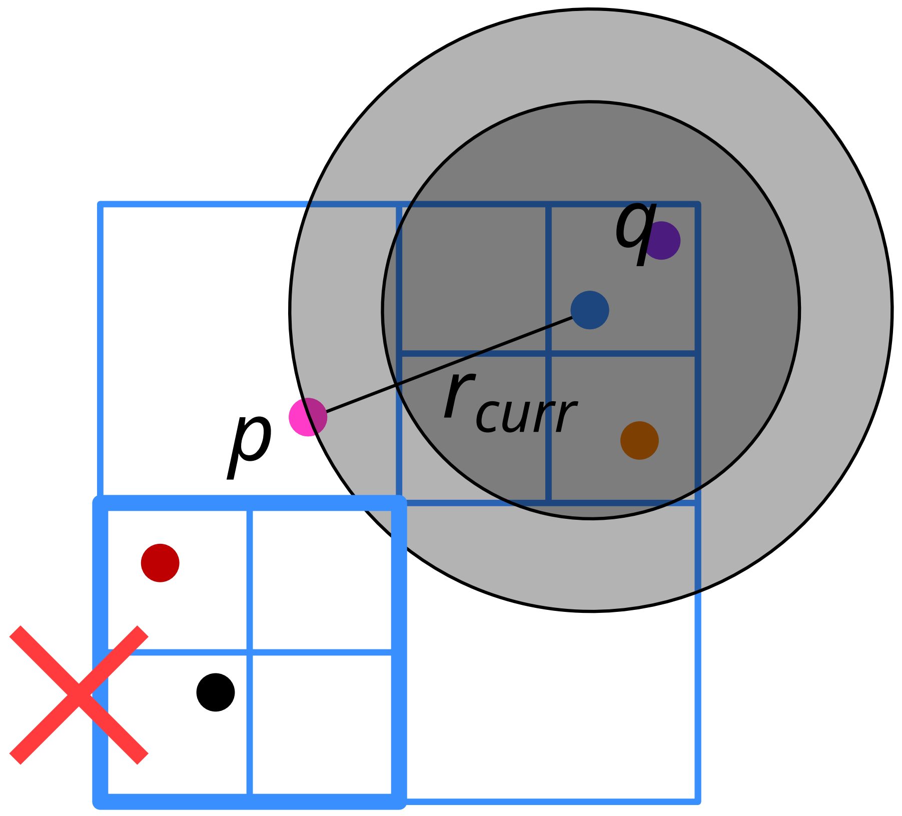
Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

only recurse on nodes/squares that could decrease distance significantly

could contain $s \in P$ with $||q - s|| < (1 - \varepsilon/2)r_{curr}$



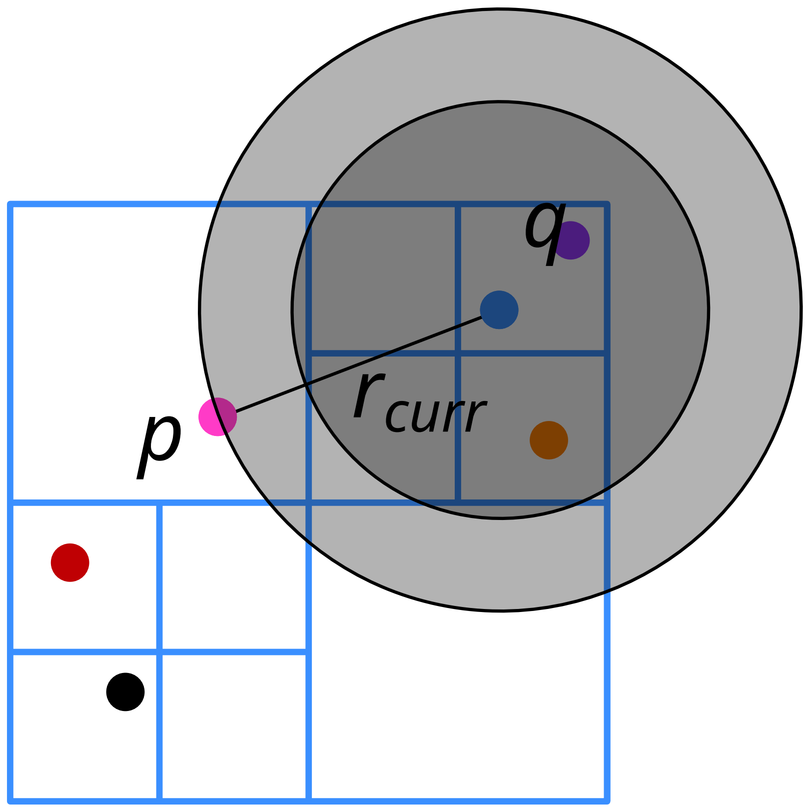
Approximate nearest neighbor (Bounded spread)

Algorithm ideas

recursive: start at root (like point location).

maintain best candidate p for nearest neighbor and distance $r_{curr} = d(q, p)$

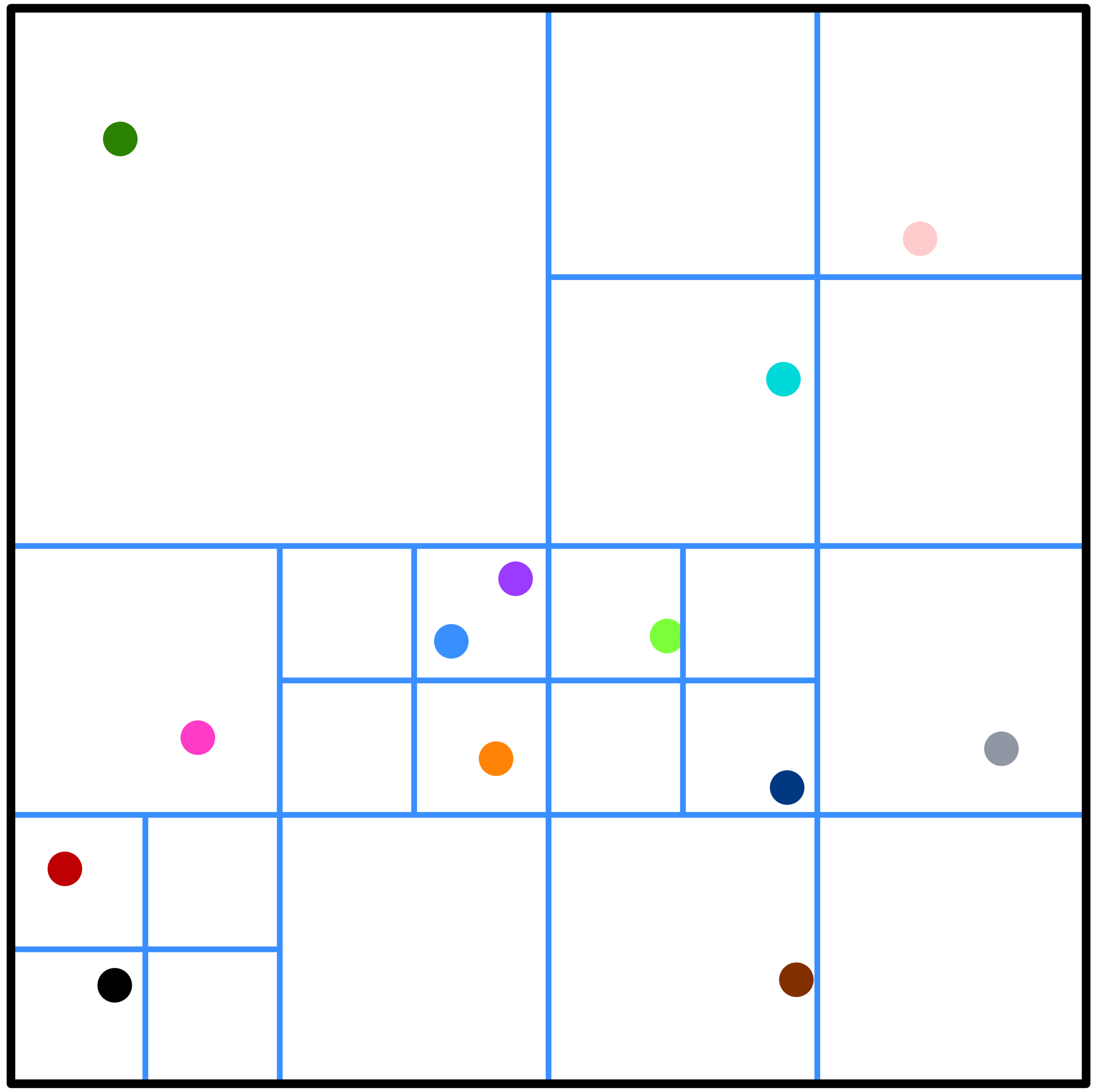
only recurse on nodes/squares that could decrease distance significantly

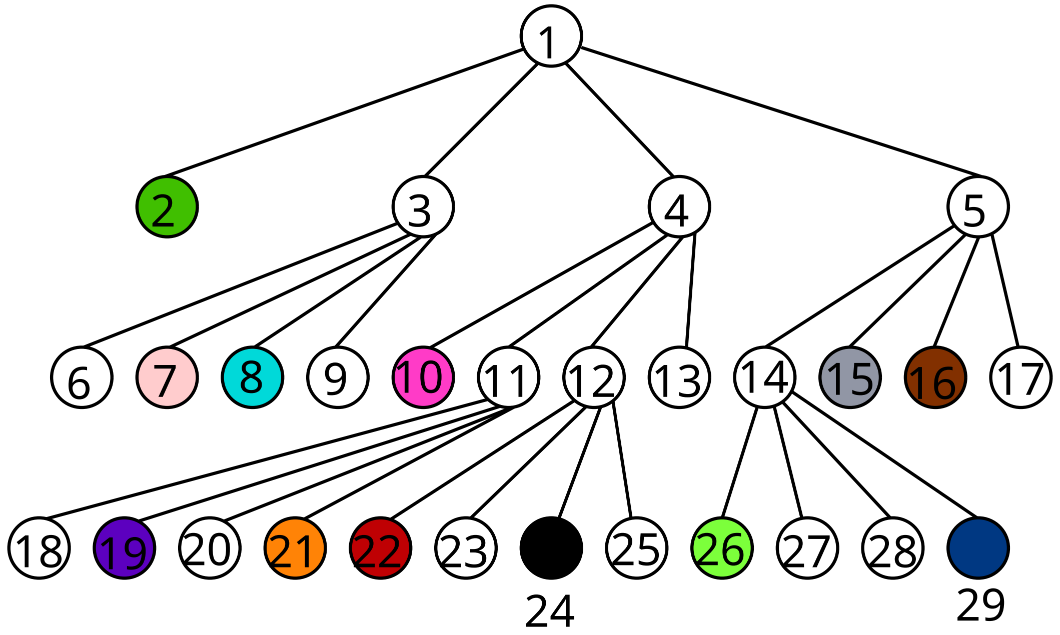


could contain $s \in P$ with $||q - s|| < (1 - \varepsilon/2)r_{curr}$

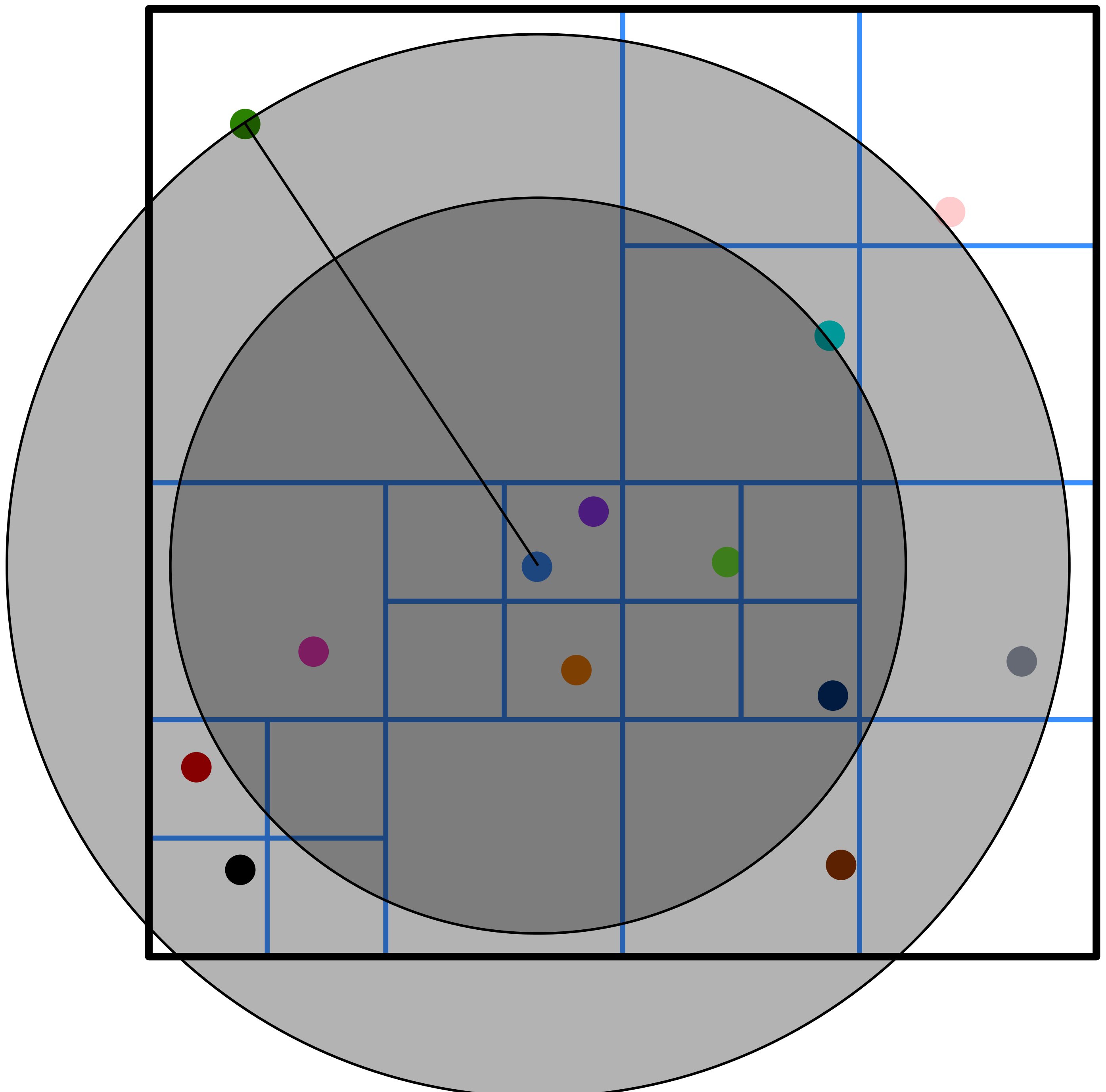
ignore cell w if $||q - \text{rep}_w|| - diam(\square_w) > (1 - \varepsilon/2)r_{curr}$

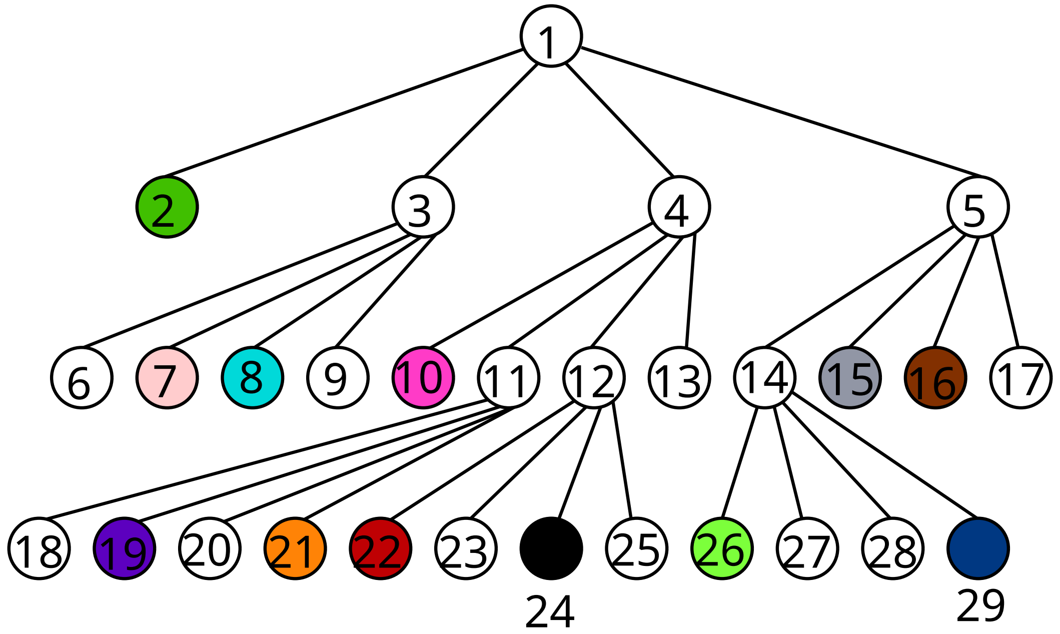
Approximate nearest neighbor (Bounded spread)





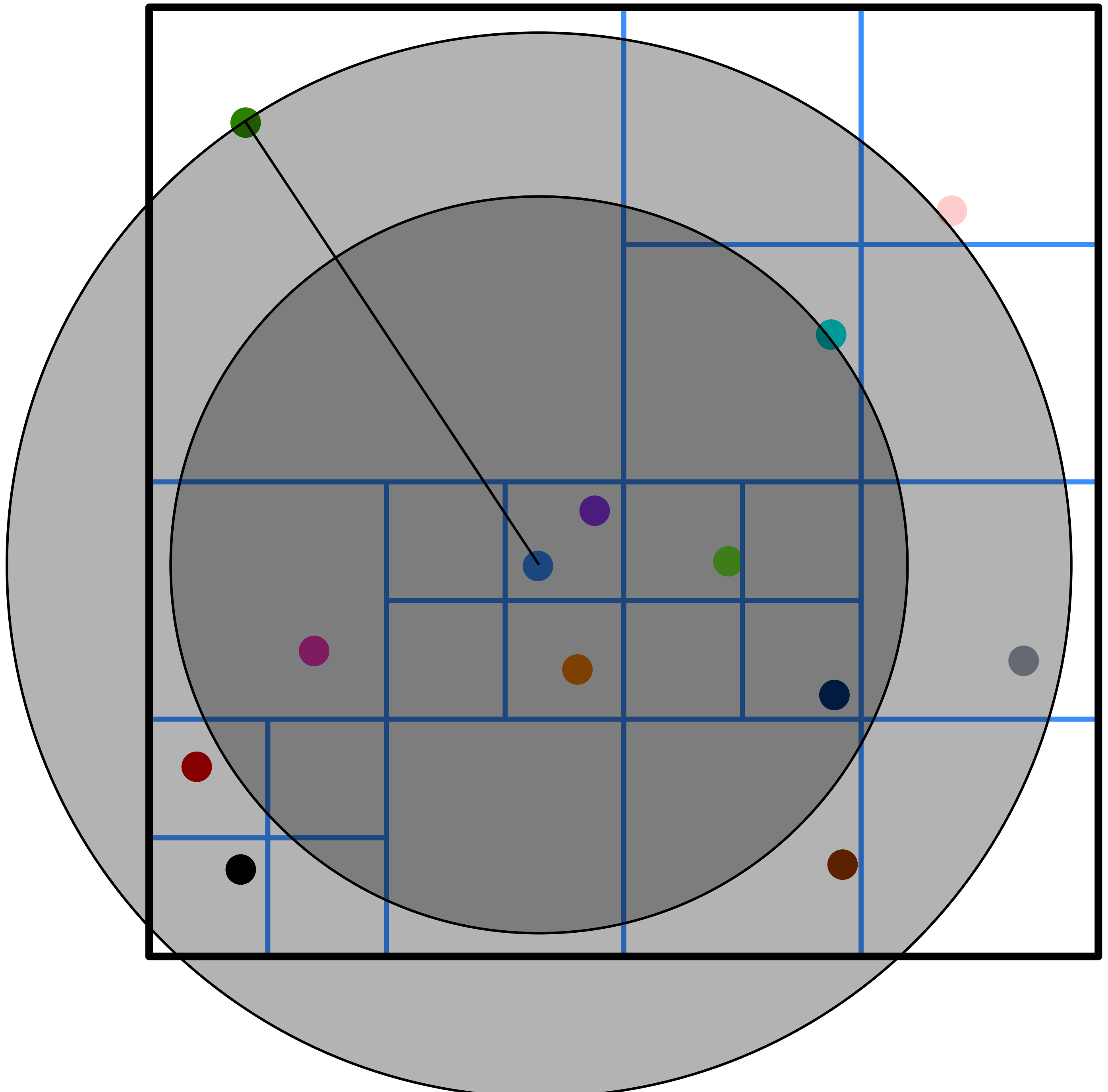
Approximate nearest neighbor (Bounded spread)

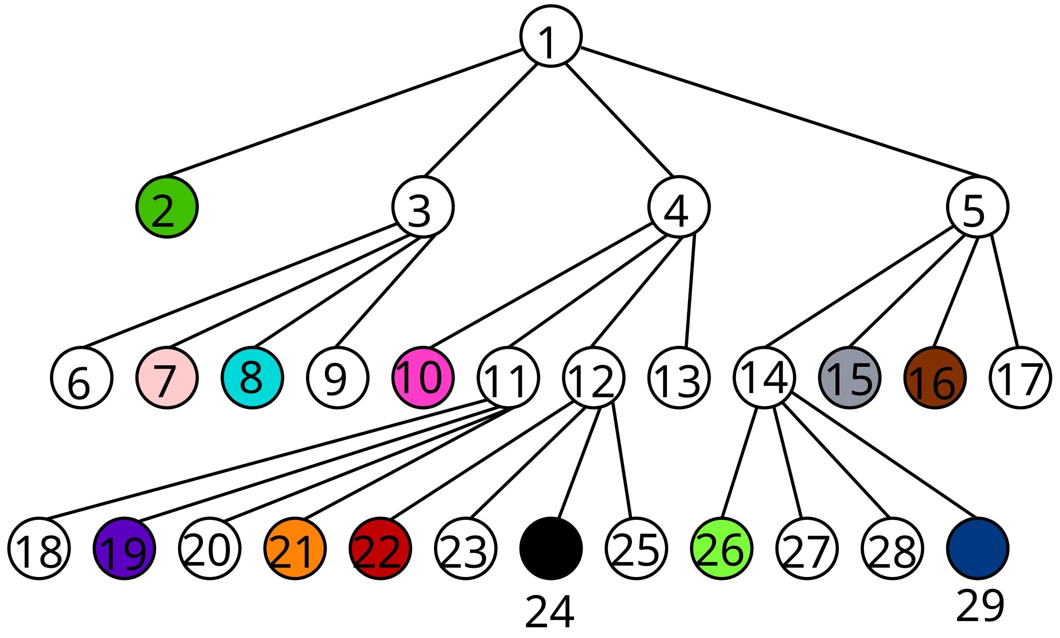




$A_0 = \{1\}$
$\text{rep}_1 = 2$
$p = 2$

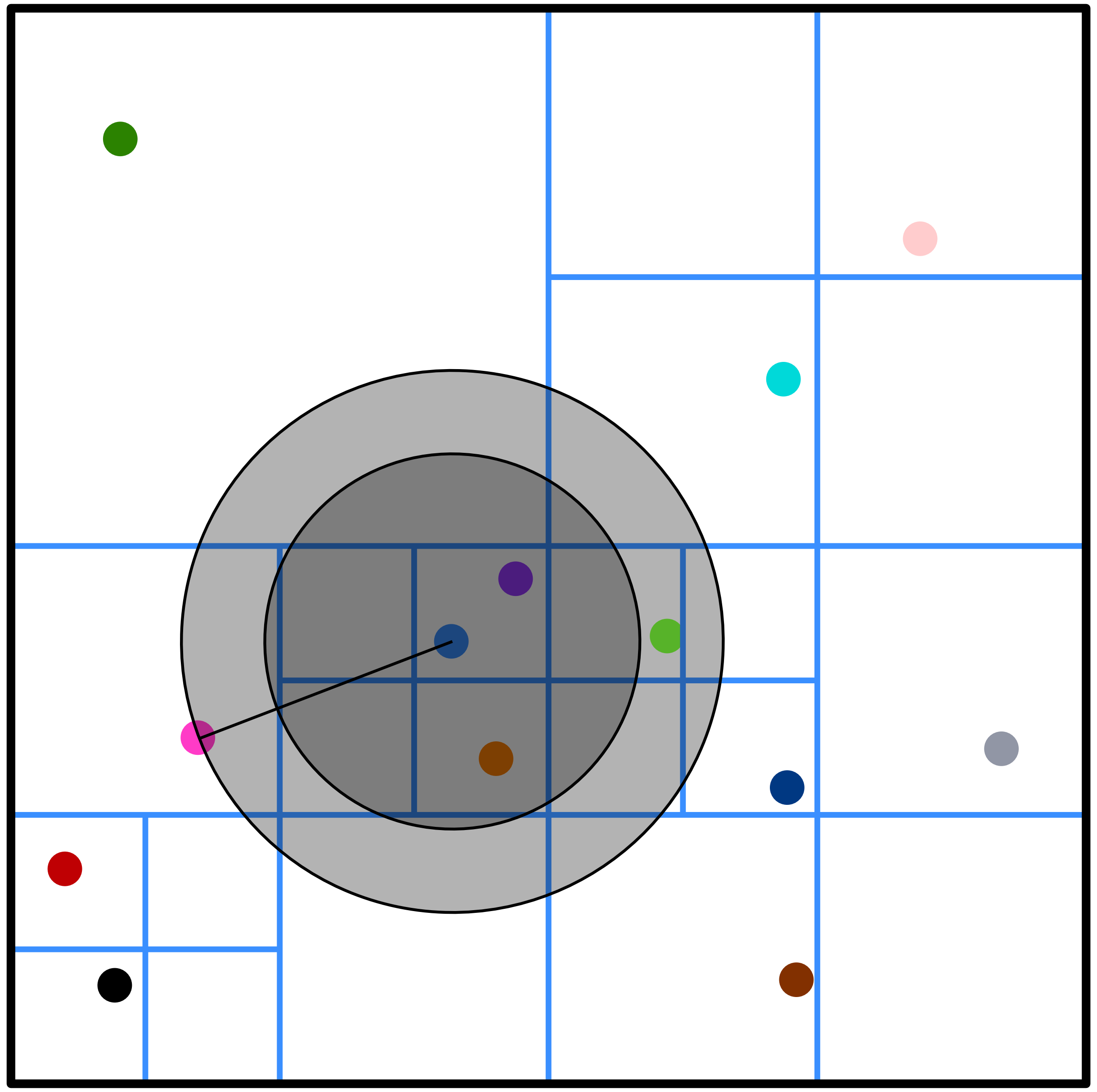
Approximate nearest neighbor (Bounded spread)

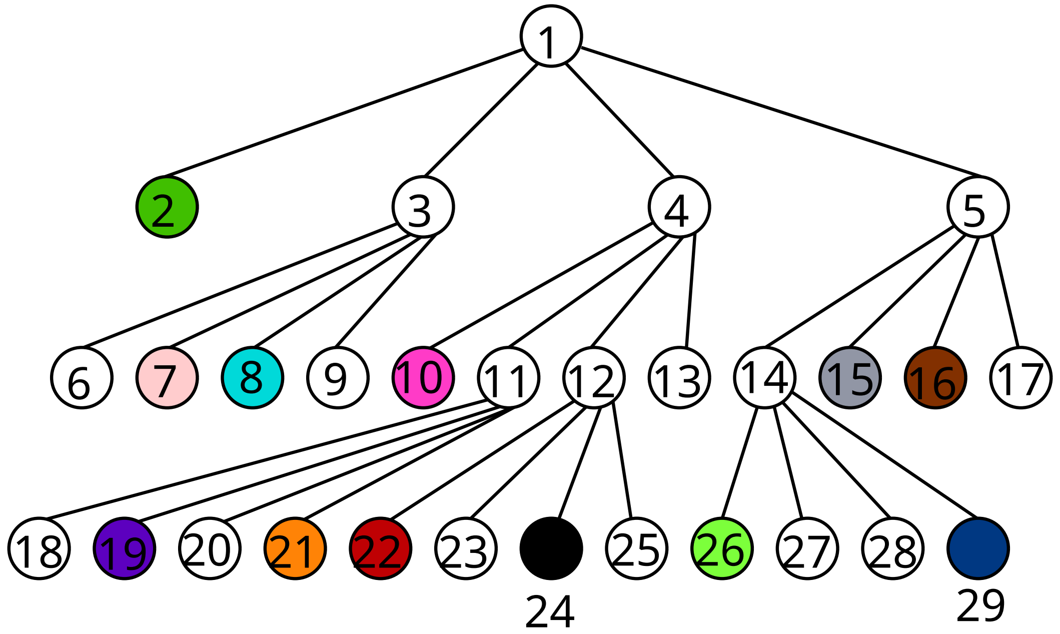




$A_1 = \{2, 3, 4, 5\}$
$\text{rep}_3 = 7$, $\text{rep}_4 = 10$, $\text{rep}_5 = 15$
$p = 2$

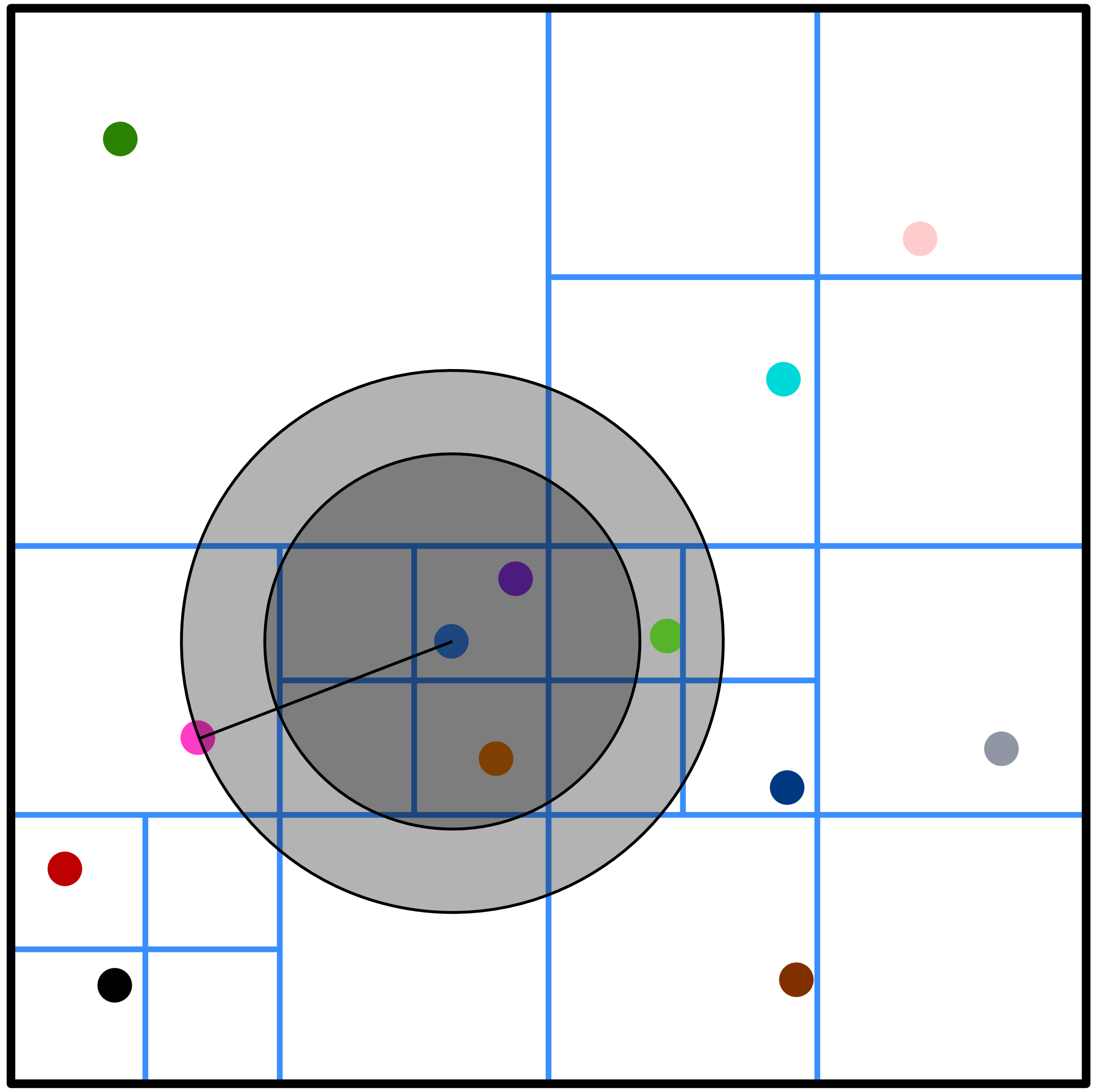
Approximate nearest neighbor (Bounded spread)

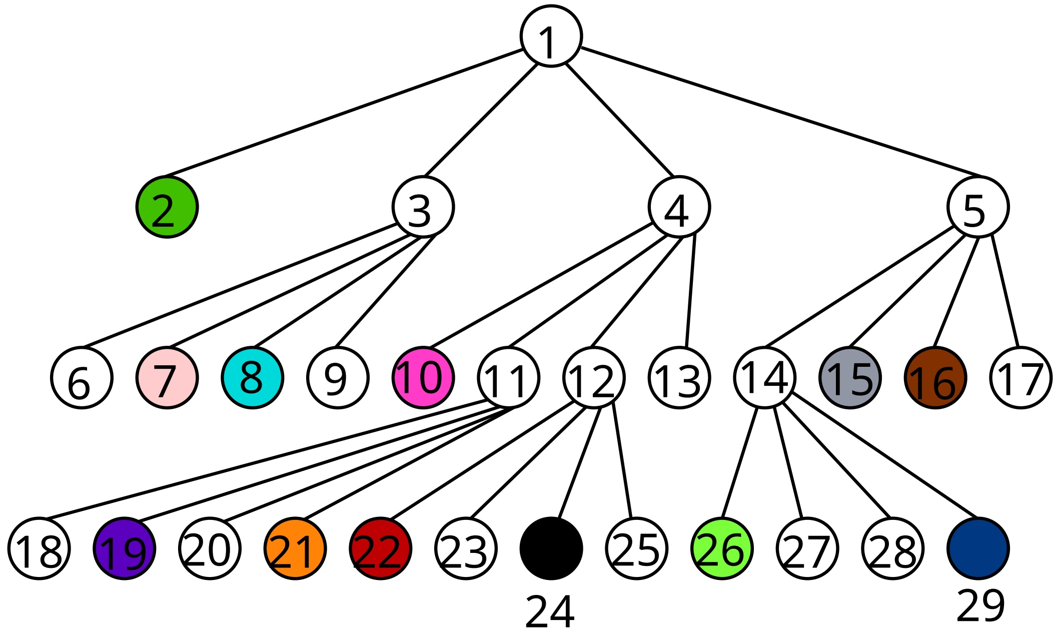




$A_1 = \{2, 3, 4, 5\}$
$\mathrm{rep}_3 = 7$, $\mathrm{rep}_4 = 10$, $\mathrm{rep}_5 = 15$
$p = 10$

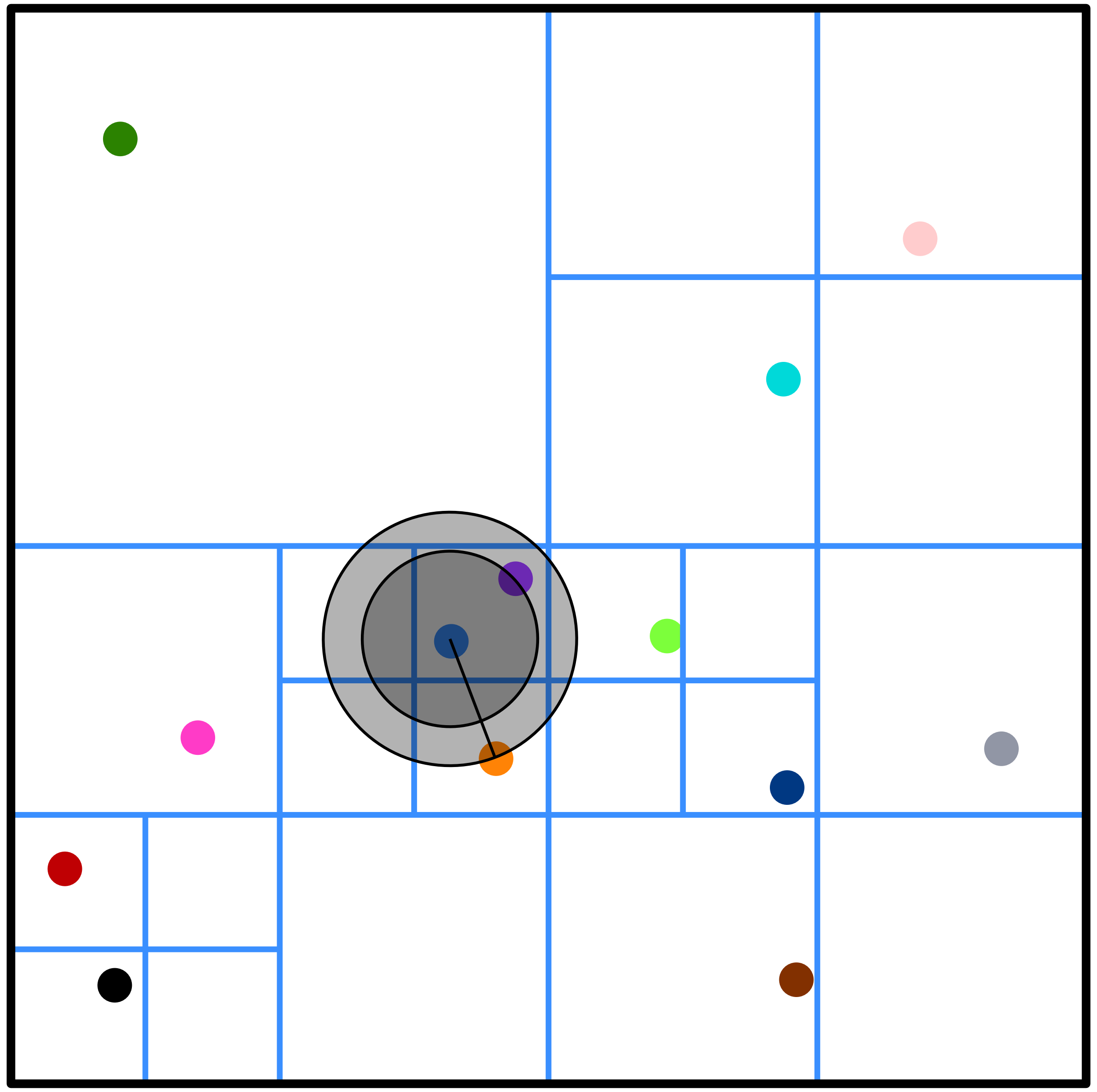
Approximate nearest neighbor (Bounded spread)

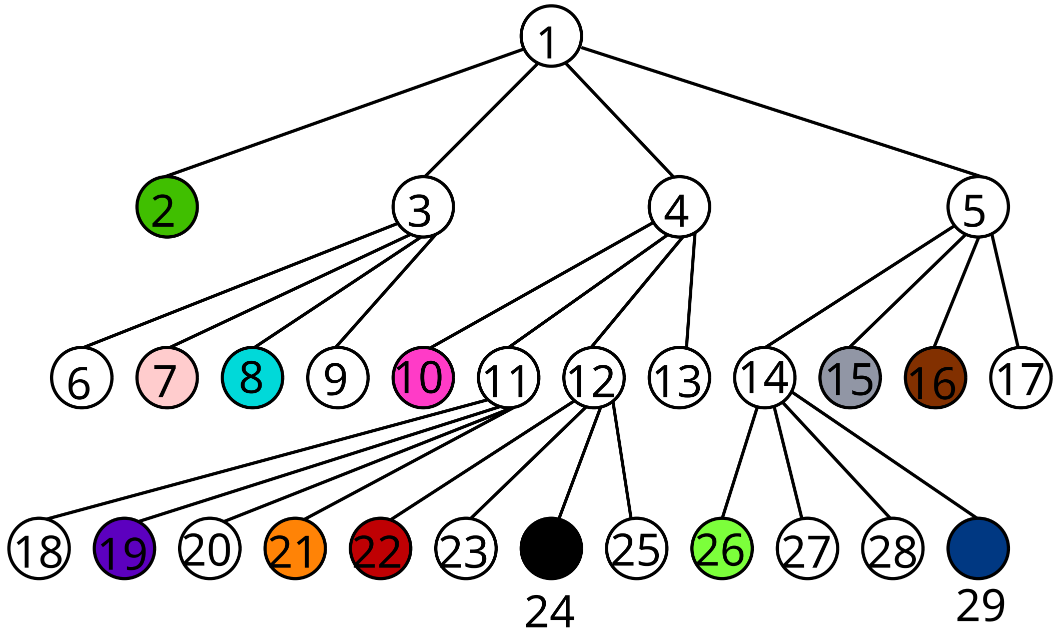




$A_2 = \{8, 10, 11, 14\}$
$\mathrm{rep}_{11} = 21, \mathrm{rep}_{14} = 26$
$p = 10$

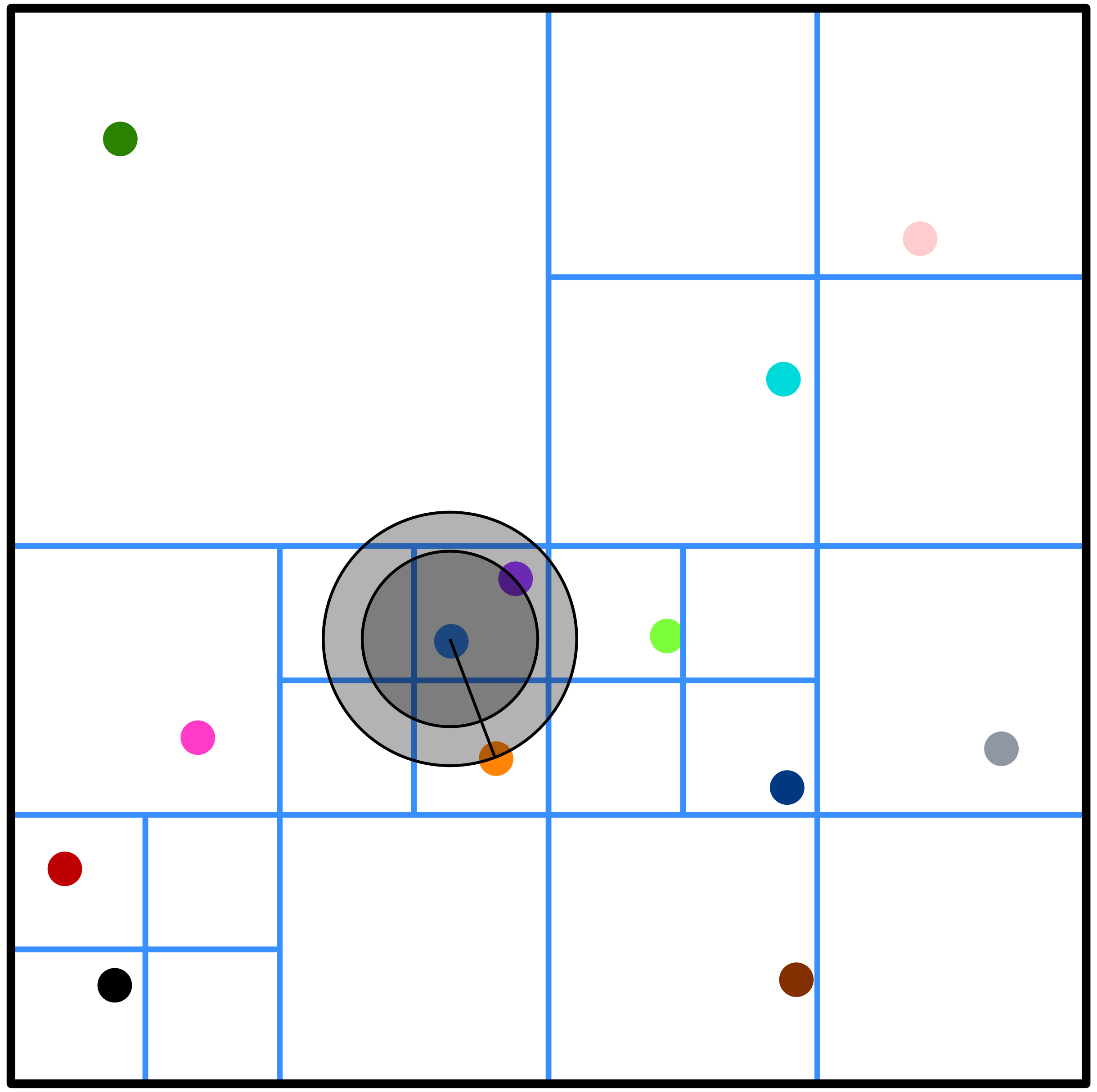
Approximate nearest neighbor (Bounded spread)

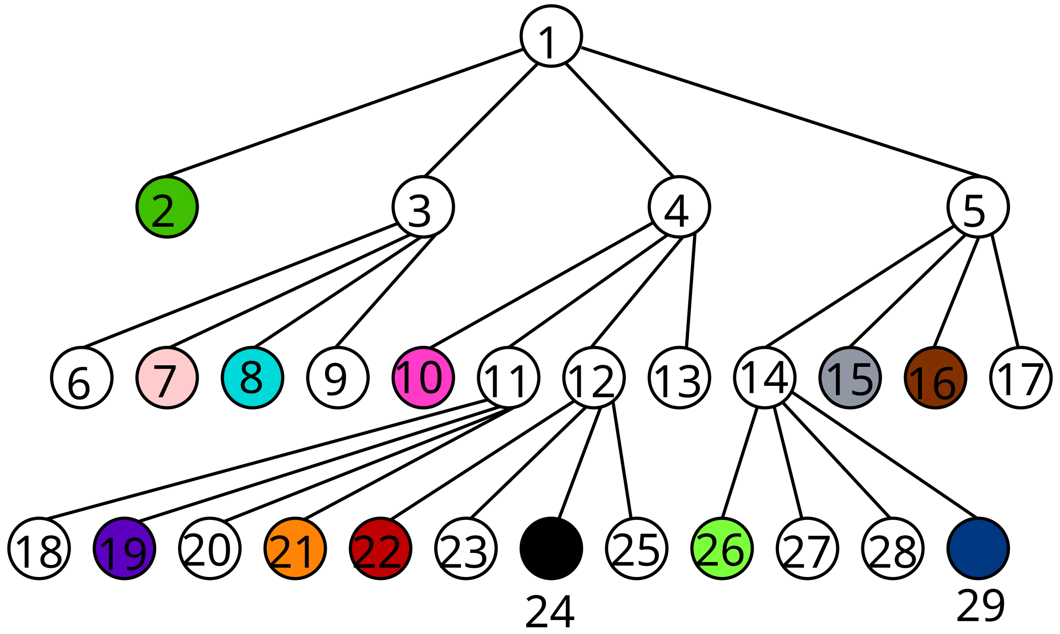




$A_2 = \{8, 10, 11, 14\}$
$\mathrm{rep}_{11} = 21$, $\mathrm{rep}_{14} = 26$
$p = 21$

Approximate nearest neighbor (Bounded spread)

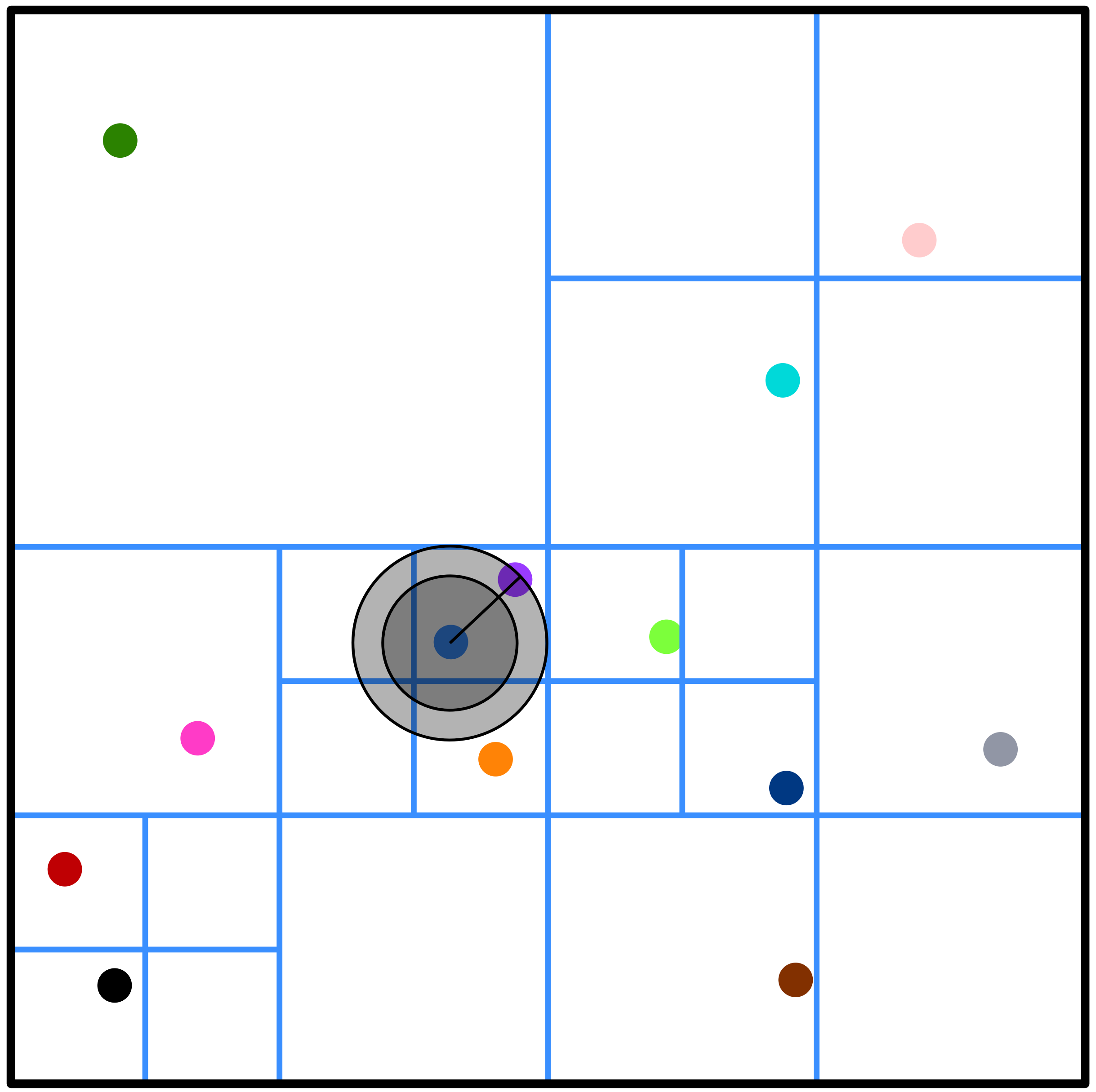


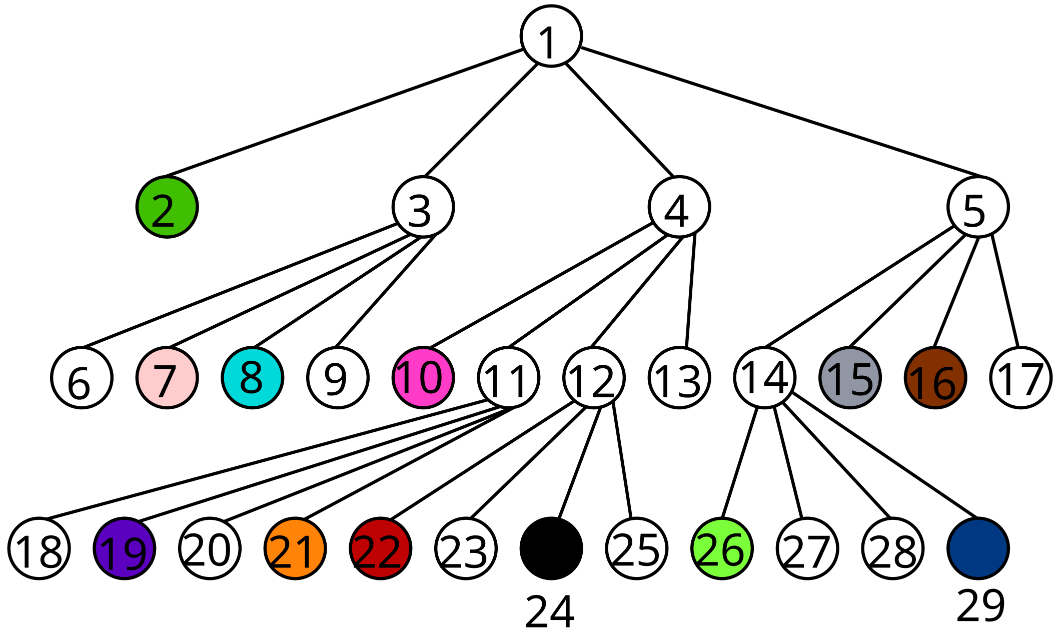


$$A_3 = \{21, 19, 26\}$$

$$p = 21$$

Approximate nearest neighbor (Bounded spread)

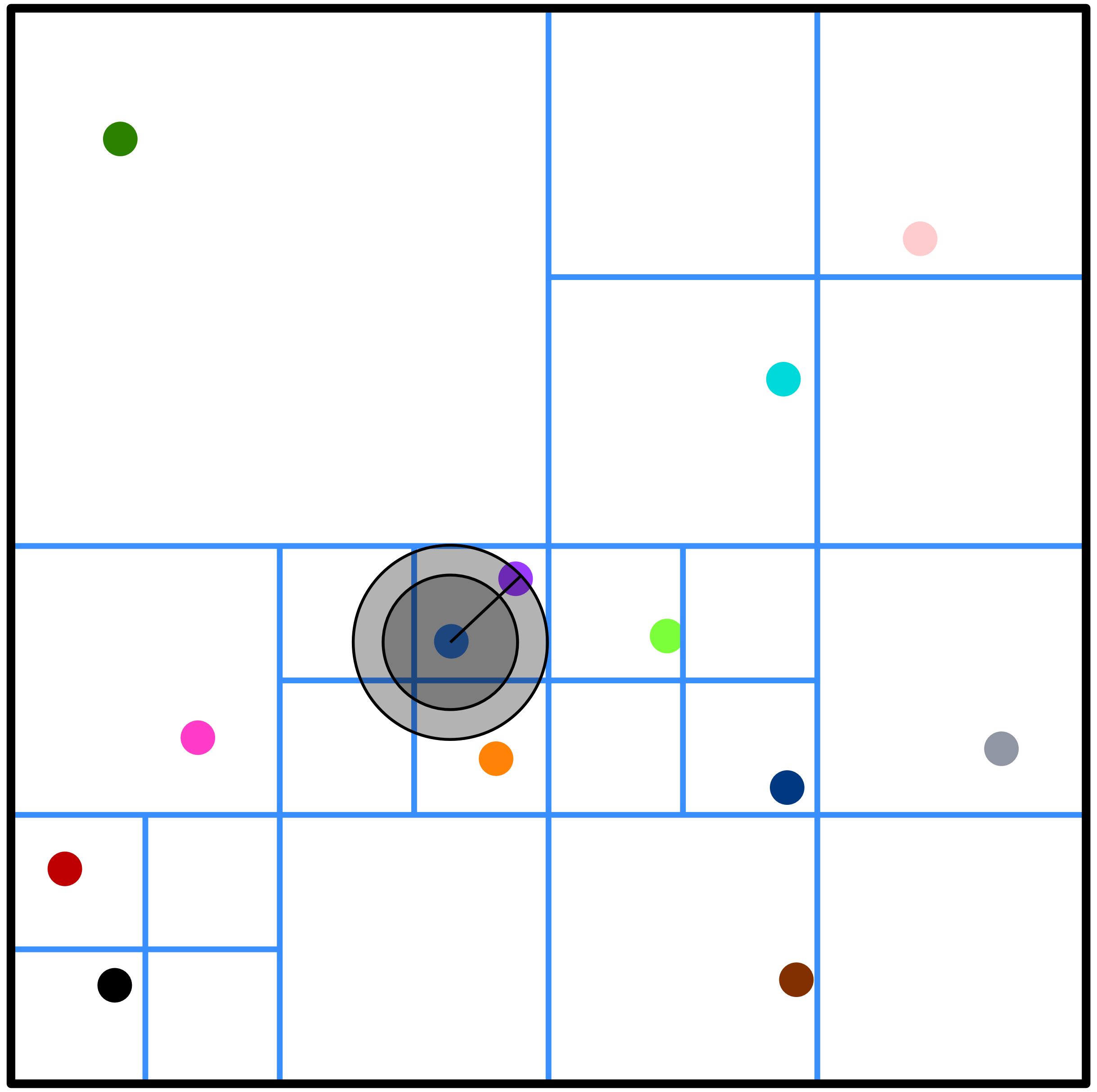


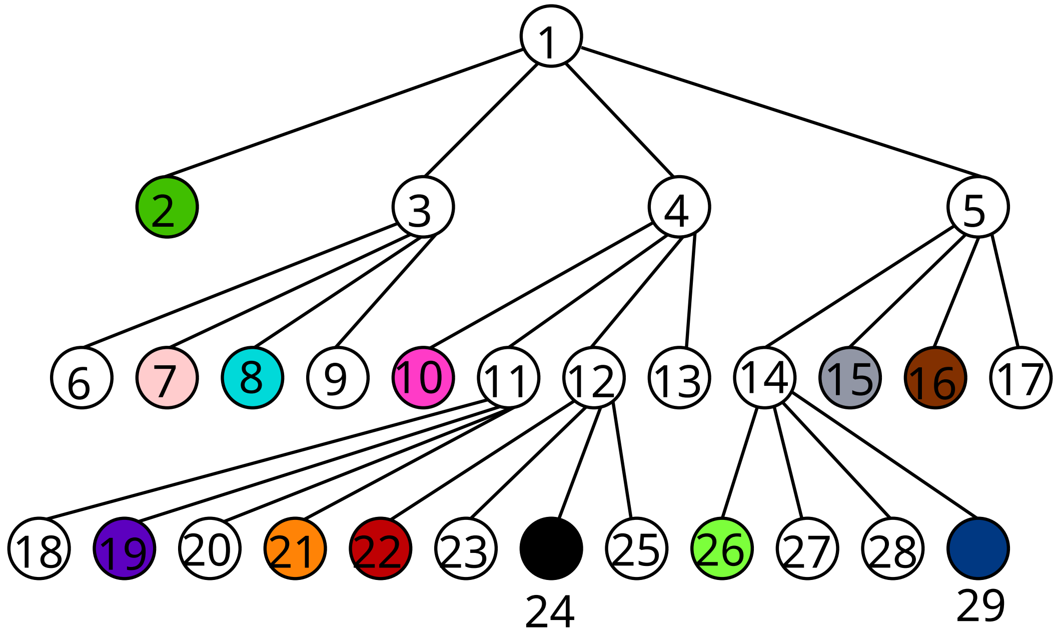


$A_3 = \{21, 19, 26\}$

$p = 19$

Approximate nearest neighbor (Bounded spread)





Question: How do we analyze the running time?

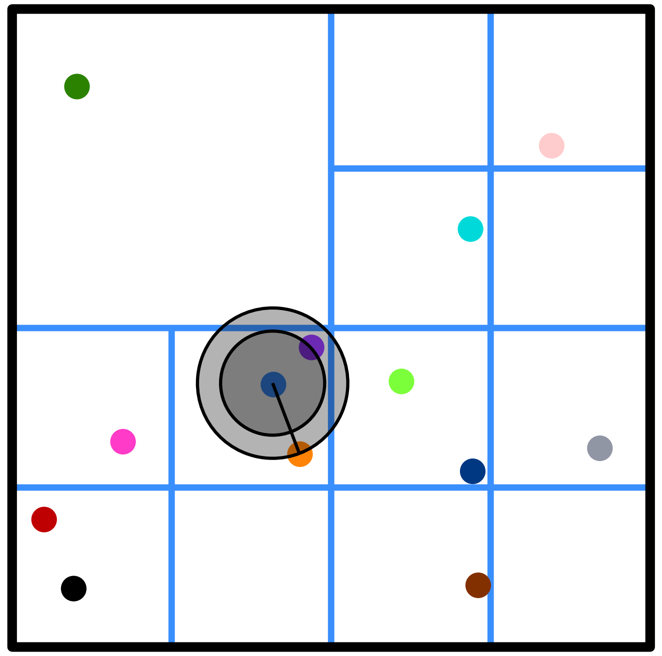
Approximate nearest neighbor (Bounded spread)

Running time main ideas

Approximate nearest neighbor (Bounded spread)

Running time main ideas

1. as long as cells > $d(q, P)$ only $O(1)$ cells per level

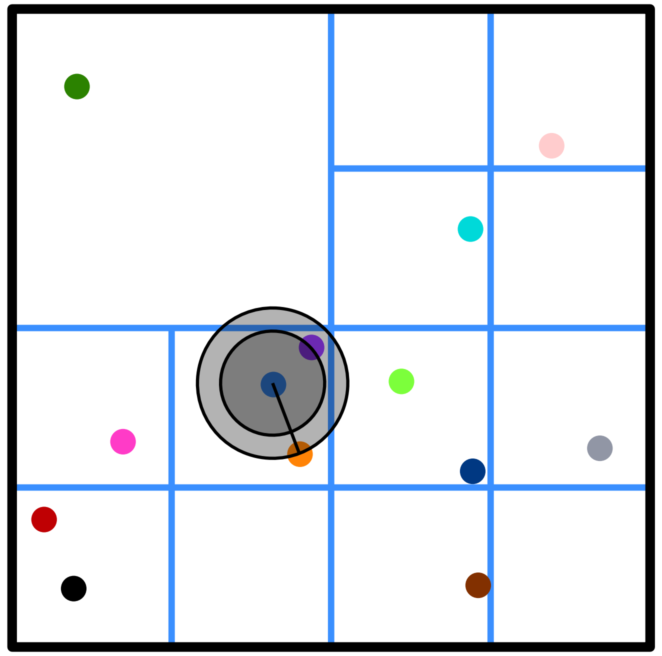


Approximate nearest neighbor (Bounded spread)

Running time main ideas

1. as long as cells > $d(q, P)$ only $O(1)$ cells per level

$$\# \text{ such cells} = O(\text{height}) = O(\log \Phi(P))$$

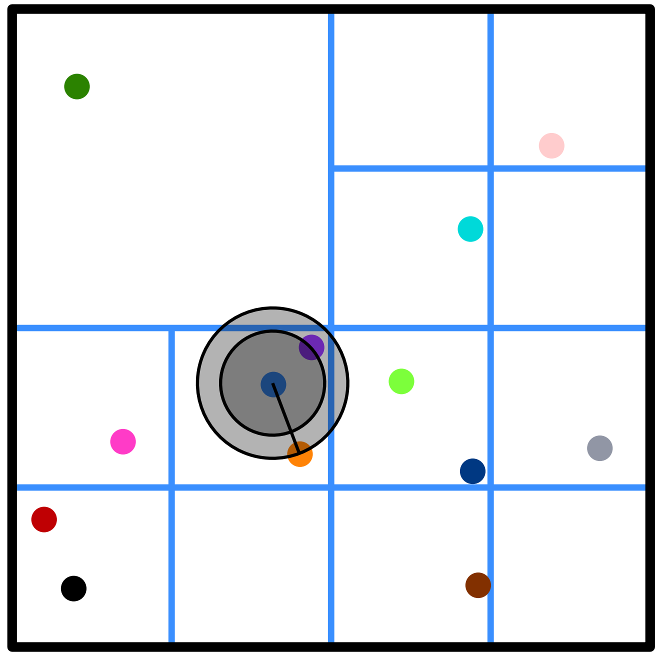


Approximate nearest neighbor (Bounded spread)

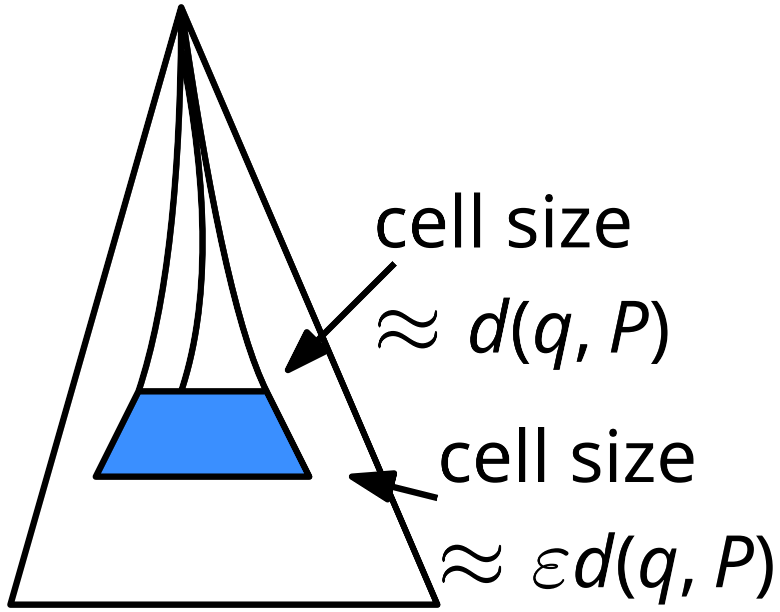
Running time main ideas

1. as long as cells > $d(q, P)$ only $O(1)$ cells per level

   \# such cells = $O(\text{height}) = O(\log \Phi(P))$



2. ends when cells have size $\varepsilon d(q, P)$

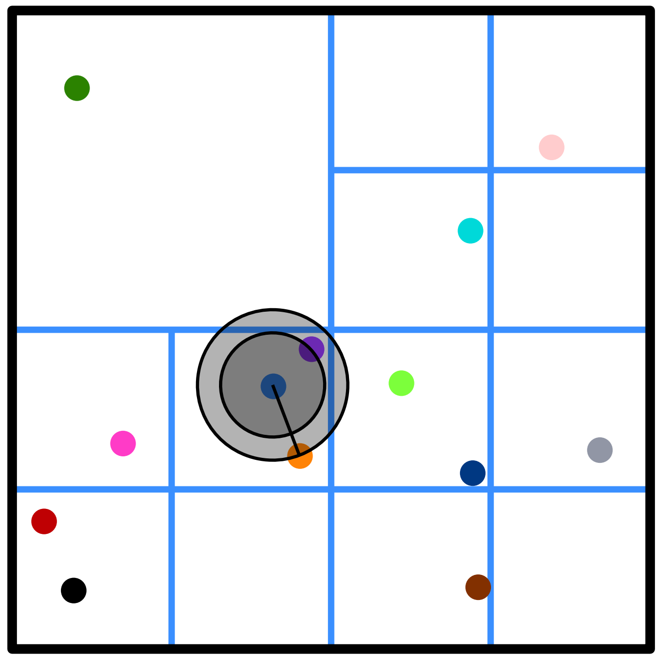


Approximate nearest neighbor (Bounded spread)

Running time main ideas

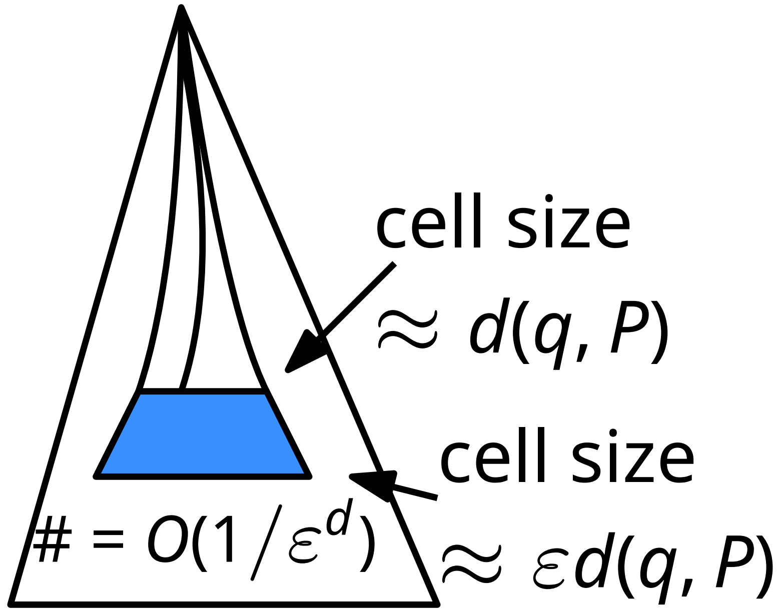
1. as long as cells > $d(q, P)$ only $O(1)$ cells per level

   # such cells = $O(\text{height})$ = $O(\log \Phi(P))$



2. ends when cells have size $\varepsilon d(q, P)$

   # cells in last levels = $O(1/\varepsilon^d)$



Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$\|q - rep_w\| - diam(\sigma_w) \geq \|q - rep_w\| - (\varepsilon/4)r$

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$$\|q - rep_w\| - diam(\sigma_w) \geq \|q - rep_w\| - (\varepsilon/4)r$$
$$\geq r_{curr} - (\varepsilon/4)r_{curr} \geq (1 - \varepsilon/4)r_{curr}$$

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$$\|q - rep_w\| - diam(\sigma_w) \geq \|q - rep_w\| - (\varepsilon/4)r$$
$$\geq r_{curr} - (\varepsilon/4)r_{curr} \geq (1 - \varepsilon/4)r_{curr}$$

side length at depth i: 2^{-i}

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$$\|q - rep_w\| - diam(\sigma_w) \geq \|q - rep_w\| - (\varepsilon/4)r$$
$$\geq r_{curr} - (\varepsilon/4)r_{curr} \geq (1 - \varepsilon/4)r_{curr}$$

side length at depth i: 2^{-i}

diameter at depth i: $\sqrt{d}2^{-i}$

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$$\begin{aligned}\|q - rep_w\| - diam(\sigma_w) &\geq \|q - rep_w\| - (\varepsilon/4)r \\ &\geq r_{curr} - (\varepsilon/4)r_{curr} \geq (1 - \varepsilon/4)r_{curr}\end{aligned}$$

side length at depth i: 2^{-i}

diameter at depth i: $\sqrt{d}2^{-i} \geq (\varepsilon/4)r$

Approximate nearest neighbor (Bounded spread)

Running time

$r := d(q, P)$

Claim: node w with square σ with $diam(\sigma) < (\varepsilon/4)r$ is not further considered

$$\|q - rep_w\| - diam(\sigma_w) \geq \|q - rep_w\| - (\varepsilon/4)r$$
$$\geq r_{curr} - (\varepsilon/4)r_{curr} \geq (1 - \varepsilon/4)r_{curr}$$

side length at depth i: 2^{-i}

diameter at depth i: $\sqrt{d}2^{-i} \geq (\varepsilon/4)r$

only levels with $i \leq -\lceil \log((\varepsilon/4)r)/\sqrt{d} \rceil$ considered

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \leq -\lceil \log((\varepsilon/4)r)\rceil$ =: h considered

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \leq -\lceil \log((\varepsilon/4)r)\rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

iteration $i + 1$: only cells at distance $r_{curr} \leq \ell_i$ to q considered.

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

iteration $i+1$: only cells at distance $r_{curr} \leq \ell_i$ to q considered.

How many? (upper bound)

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \leq -\lceil \log((\varepsilon/4)r)\rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

iteration $i + 1$: only cells at distance $r_{curr} \leq \ell_i$ to q considered.

How many? (upper bound)

at most $n_i = \left(2\lceil \frac{\ell_i}{2^{-i-1}} \rceil\right)^d$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

iteration $i+1$: only cells at distance $r_{curr} \leq \ell_i$ to q considered.

How many? (upper bound)

at most $n_i = \left(2\left\lceil \frac{\ell_i}{2^{-i-1}} \right\rceil\right)^d = O\left(\left(1 + \frac{r+\sqrt{d}2^{-i}}{2^{-i-1}}\right)^d\right)$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered

Let u be node of depth i containing $nn(q)$

$\ell_i := d(q, rep_u) \leq diam_u + r \Rightarrow$ after iteration i: $r_{curr} \leq diam_u + r = r + \sqrt{d}2^{-i}$

iteration $i+1$: only cells at distance $r_{curr} \leq \ell_i$ to q considered.

How many? (upper bound)

at most $n_i = \left(2\left\lceil \frac{\ell_i}{2^{-i-1}} \right\rceil\right)^d = O\left(\left(1 + \frac{r+\sqrt{d}2^{-i}}{2^{-i-1}}\right)^d\right)$

$$= O\left(1 + \left(2^i r\right)^d\right)$$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \leq -\lceil \log((\varepsilon/4)r)\rceil =: h$ considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil =: h$ considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

$$\sum_{i=0}^{h} n_i = O(h + 2^h r) = O(-\log(\varepsilon r) + 1/\varepsilon^d)$$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \leq -\lceil \log((\varepsilon/4)r)\rceil =: h$ considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

$$\sum_{i=0}^{h} n_i = O(h + 2^h r) = O(-\log(\varepsilon r) + 1/\varepsilon^d)$$

$$= O(\log(1/r) + 1/\varepsilon^d)$$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil =: h$ considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

$$\sum_{i=0}^{h} n_i = O(h + 2^h r) = O(-\log(\varepsilon r) + 1/\varepsilon^d)$$
$$= O(\log(1/r) + 1/\varepsilon^d)$$

Alternative bound on i: $i \leq \log \Phi(P)$

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \leq \lceil -\log((\varepsilon/4)r)/\sqrt{d} \rceil \leq -\lceil \log((\varepsilon/4)r) \rceil$ =: h considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

$$\sum_{i=0}^{h} n_i = O(h + 2^h r) = O(-\log(\varepsilon r) + 1/\varepsilon^d)$$
$$= O(\log(1/r) + 1/\varepsilon^d)$$

Alternative bound on i: $i \leq \log \Phi(P)$

Summary:

A $(1 + \varepsilon)$-ANN query on a quadtree takes $O(1/\varepsilon^d + \log \Phi(P))$ time.

Approximate nearest neighbor (Bounded spread)

Running time

- only levels with $i \le \lceil -\log((\varepsilon/4)r)/\sqrt{d}\rceil \le -\lceil \log((\varepsilon/4)r)\rceil$ =: h considered
- cells further considered at depth i: $n_i = O\left(1 + \left(2^i r\right)^d\right)$

$$\sum_{i=0}^{h} n_i = O(h + 2^h r) = O(-\log(\varepsilon r) + 1/\varepsilon^d)$$
$$= O(\log(1/r) + 1/\varepsilon^d)$$

Alternative bound on i: $i \le \log \Phi(P)$

Summary:

A $(1+\varepsilon)$-ANN query on a quadtree takes $O(1/\varepsilon^d + \log \Phi(P))$ time.

How about unbounded spread?

Overview

1. Introduction
2. ANN with quadtree (bounded spread)
3. Why low-quality approximation helps for unbounded spread
4. Low-quality approximation

low-quality approximation $\rightarrow$ unbounded spread

Assume we can compute p that is $4n$-ANN of q.

low-quality approximation $\to$ unbounded spread

Assume we can compute p that is $4n$-ANN of q.

$R := \|p - q\|$, $L := \lfloor \log R \rfloor$

Algorithm

1. Compute $4n$-approximation

2. Find cells of grid G_{2^L} at distance $\leq R$ from q

3. Use algorithm for bounded spread (extended to compressed quadtrees) on these cells

low-quality approximation $\to$ unbounded spread

Assume we can compute p that is $4n$-ANN of q.

$R := \|p - q\|$, $L := \lfloor \log R \rfloor$

Algorithm

1. Compute $4n$-approximation

2. Find cells of grid G_{2^L} at distance $\leq R$ from q

3. Use algorithm for bounded spread (extended to compressed quadtrees) on these cells

in short:
running time (without step 1) = $O(1/\varepsilon^2 + \log(R/r)) = O(1/\varepsilon^2 + \log n)$

Overview

1. Introduction
2. ANN with quadtree (bounded spread)
3. Why low-quality approximation helps for unbounded spread
4. Low-quality approximation:

    ring separator tree

    later lecture: shifting

Low-quality approximate nearest neighbour search

The Ring Separator Tree

A binary tree T having the points of P as leaves is a *t-ring tree* for P iff:

The Ring Separator Tree

A binary tree T having the points of P as leaves is a *t-ring tree* for P iff:

- Every node $v \in T$, with corresponding subset $P_v \subseteq P$ is associated with a 'ring' that separates the points of P_v into two sets

The Ring Separator Tree

A binary tree T having the points of P as leaves is a *t-ring tree* for P iff:

- Every node $v \in T$, with corresponding subset $P_v \subseteq P$ is associated with a 'ring' that separates the points of P_v into two sets
- The interior of the ring has no points inside it

The Ring Separator Tree

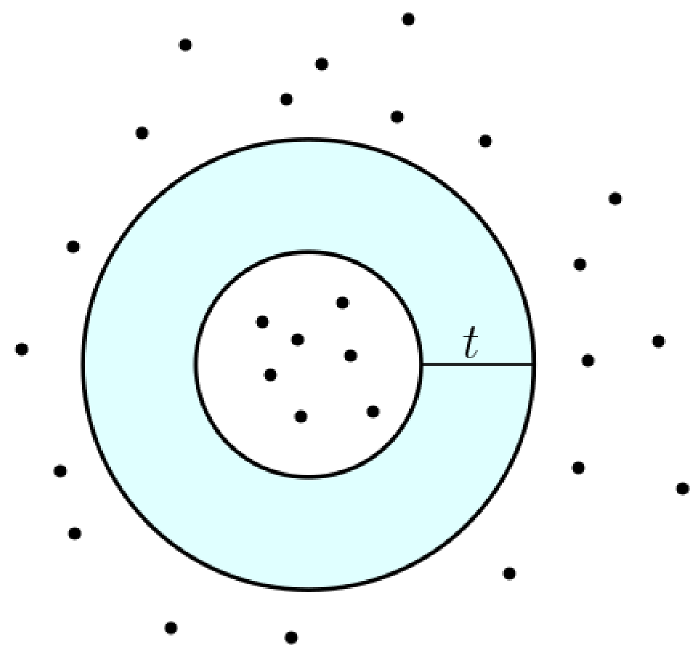
A binary tree T having the points of P as leaves is a *t-ring tree* for P iff:

- Every node $v \in T$, with corresponding subset $P_v \subseteq P$ is associated with a 'ring' that separates the points of P_v into two sets
- The interior of the ring has no points inside it
- The interior of the ring is of width t

The Ring Separator Tree

A binary tree T having the points of P as leaves is a *t-ring tree* for P iff:

- Every node $v \in T$, with corresponding subset $P_v \subseteq P$ is associated with a 'ring' that separates the points of P_v into two sets
- The interior of the ring has no points inside it
- The interior of the ring is of width *t*



The Ring Separator Tree

For every node v we ensure the following:

The Ring Separator Tree

For every node v we ensure the following:

- There is a ball $b_v = b(c_v, r_v)$ such that all points of such that all the points of $P_v^{in} = P_v \cap b_v$ are in one child of v (the 'inner' child)

The Ring Separator Tree

For every node *v* we ensure the following:

- There is a ball $b_v = b(c_v, r_v)$ such that all the points of $P_v^{in} = P_v \cap b_v$ are in one child of *v* (the 'inner' child)
- All other points of P_v are outside the enlarged ball $b(c_v, (1+t) \cdot r_v)$ and are stored in the other child (the 'outer' child)

The Ring Separator Tree

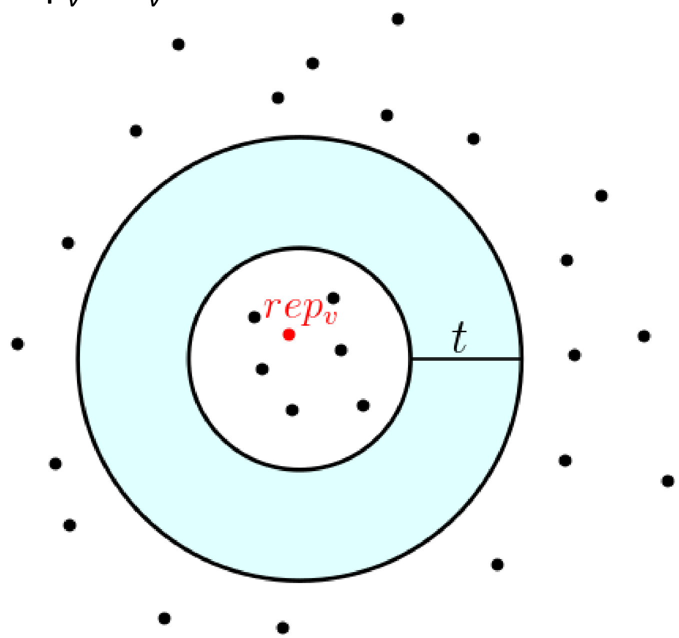
For every node v we ensure the following:

- There is a ball $b_v = b(c_v, r_v)$ such that all the points of $P_v^{in} = P_v \cap b_v$ are in one child of v (the 'inner' child)
- All other points of P_v are outside the enlarged ball $b(c_v, (1+t) \cdot r_v)$ and are stored in the other child (the 'outer' child)
- We store an arbitrary $rep_v \in P_v^{in}$ in v

The Ring Separator Tree

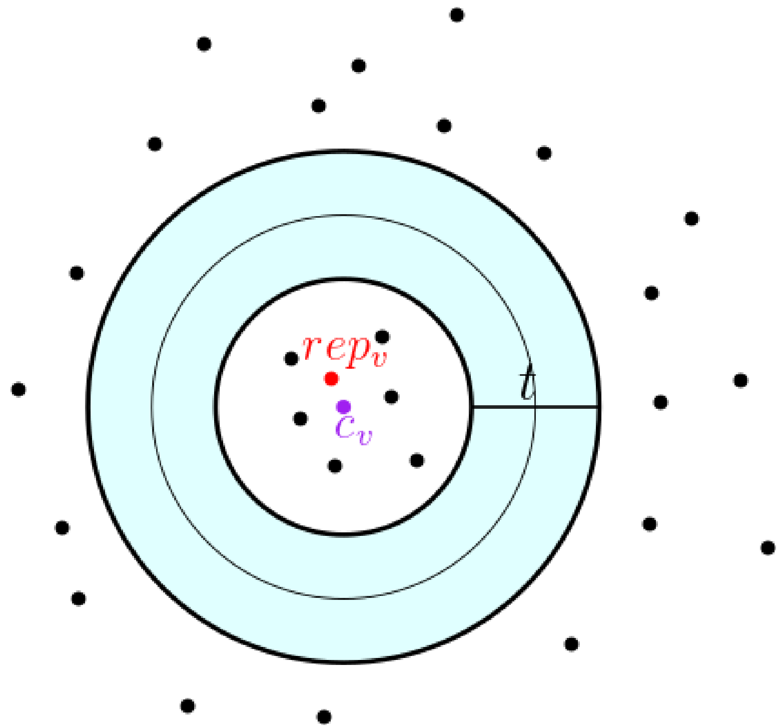
For every node v we ensure the following:

- There is a ball $b_v = b(c_v, r_v)$ such that all the points of $P_v^{in} = P_v \cap b_v$ are in one child of v (the 'inner' child)
- All other points of P_v are outside the enlarged ball $b(c_v, (1+t)\cdot r_v)$ and are stored in the other child (the 'outer' child)
- We store an arbitrary $rep_v \in P_v^{in}$ in v



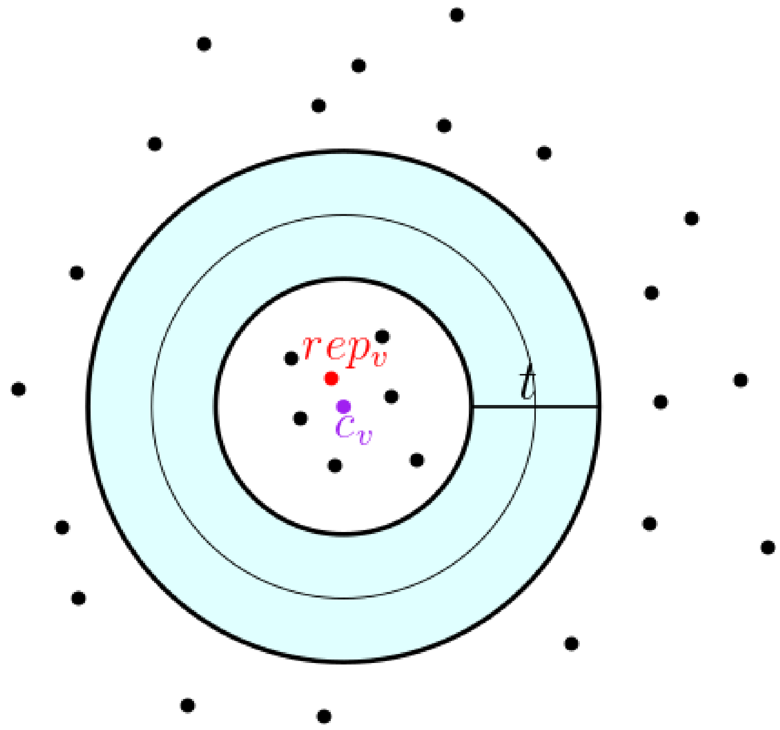
Search procedure

Given query point q:



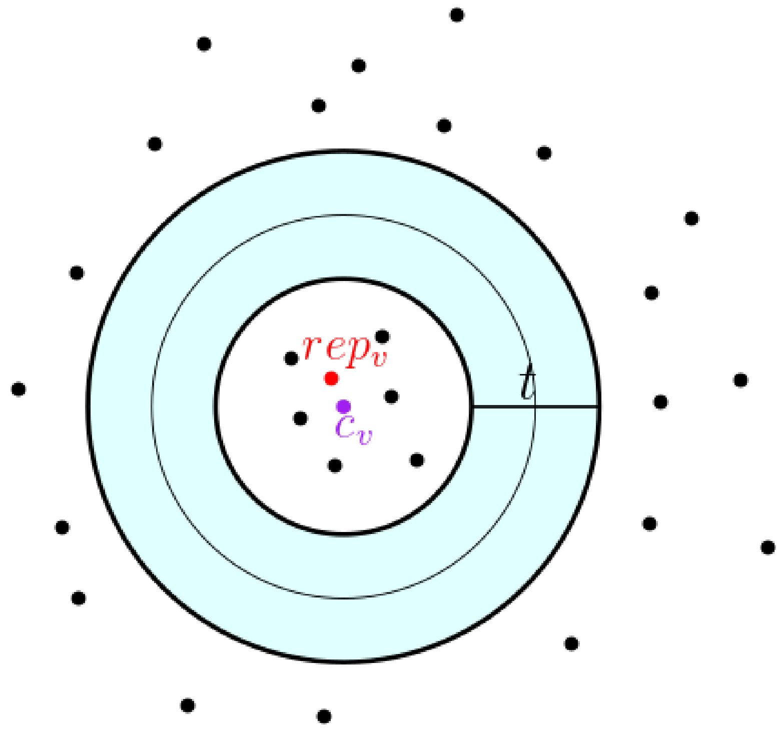
Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$



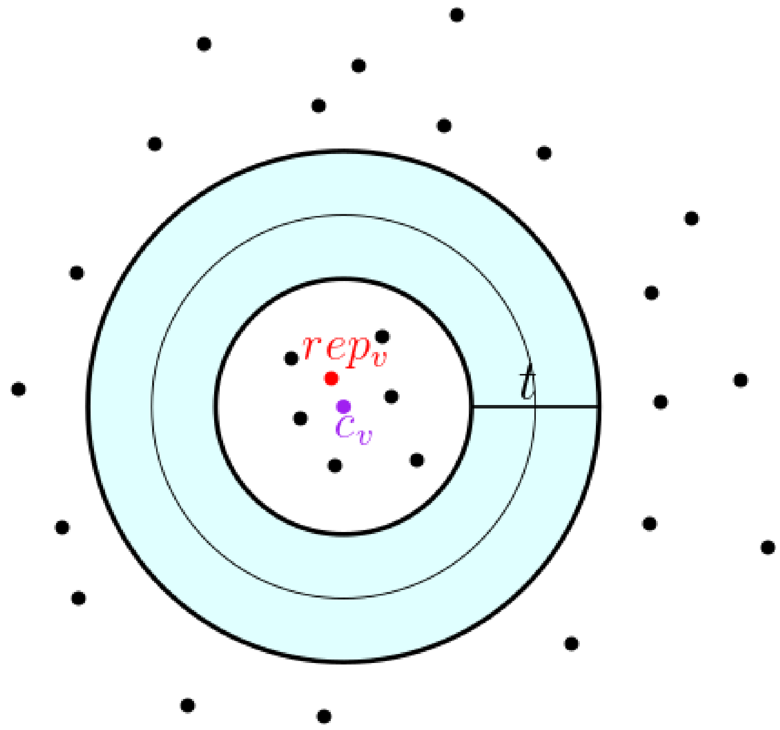
Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:



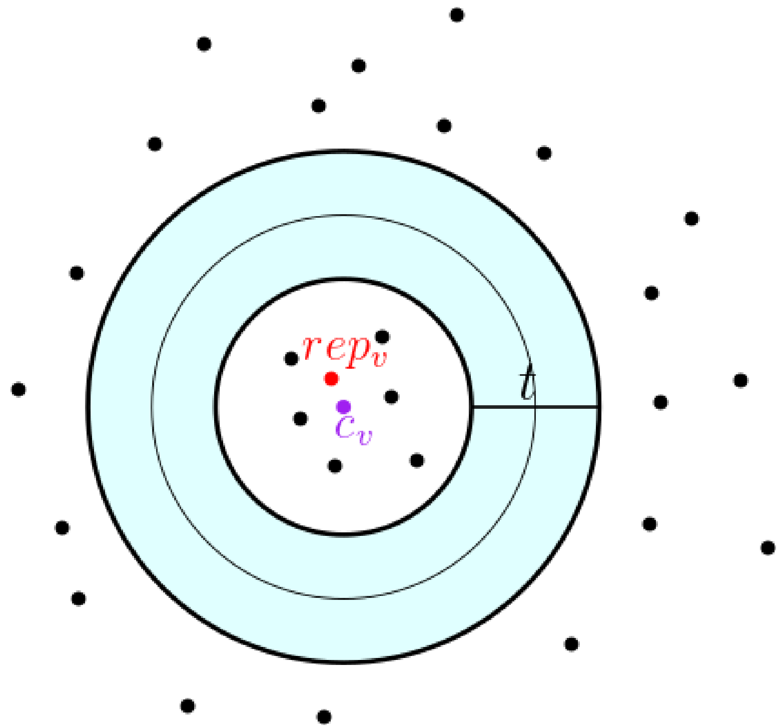
Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:
    $L \leftarrow ||q - rep_v||$



Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:
  $L \leftarrow ||q - rep_v||$
  if $L < r_{curr}$ then $r_{curr} \leftarrow L$



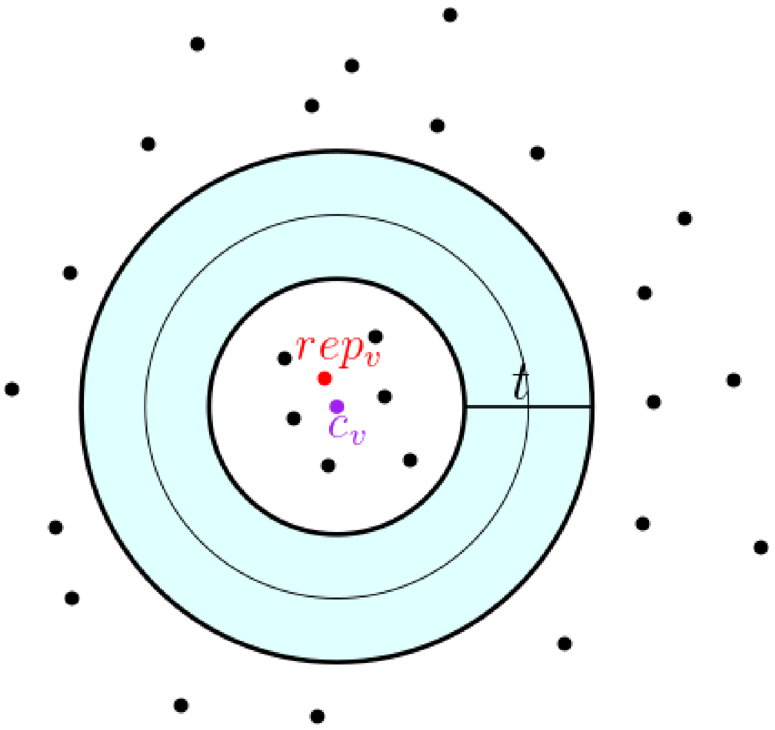
Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:

$L \leftarrow ||q - rep_v||$

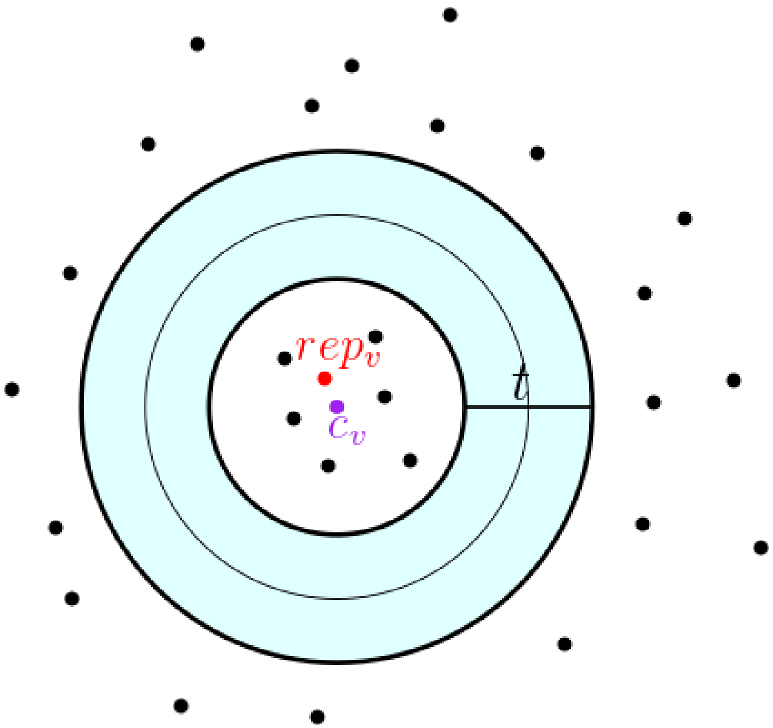
if $L < r_{curr}$ then $r_{curr} \leftarrow L$

$r_{mid} \leftarrow (1 + t/2) \cdot r_v$



Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:
$\quad L \leftarrow ||q - rep_v||$
$\quad$ if $L < r_{curr}$ then $r_{curr} \leftarrow L$
$\quad r_{mid} \leftarrow (1 + t/2) \cdot r_v$
$\quad$ if $||q - c_v|| \leq r_{mid}$ then
$\quad\quad v \leftarrow$ 'inner' child of v



Search procedure

Given query point q:

$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$

while v is not a leaf:

$\quad L \leftarrow ||q - rep_v||$

$\quad$ if $L < r_{curr}$ then $r_{curr} \leftarrow L$

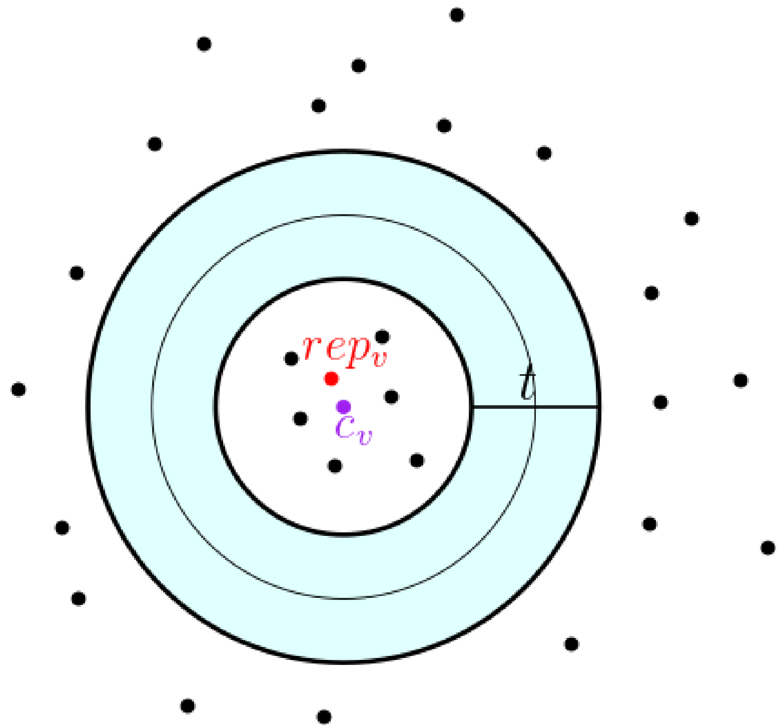
$\quad r_{mid} \leftarrow (1 + t/2) \cdot r_v$

$\quad$ if $||q - c_v|| \leq r_{mid}$ then

$\quad\quad v \leftarrow$ 'inner' child of v

$\quad$ else

$\quad\quad v \leftarrow$ 'outer' child of v



Search procedure

Given query point q:
$v \leftarrow$ root of T, $r_{curr} \leftarrow \infty$
while v is not a leaf:

$L \leftarrow ||q - rep_v||$

if $L < r_{curr}$ then $r_{curr} \leftarrow L$

$r_{mid} \leftarrow (1 + t/2) \cdot r_v$

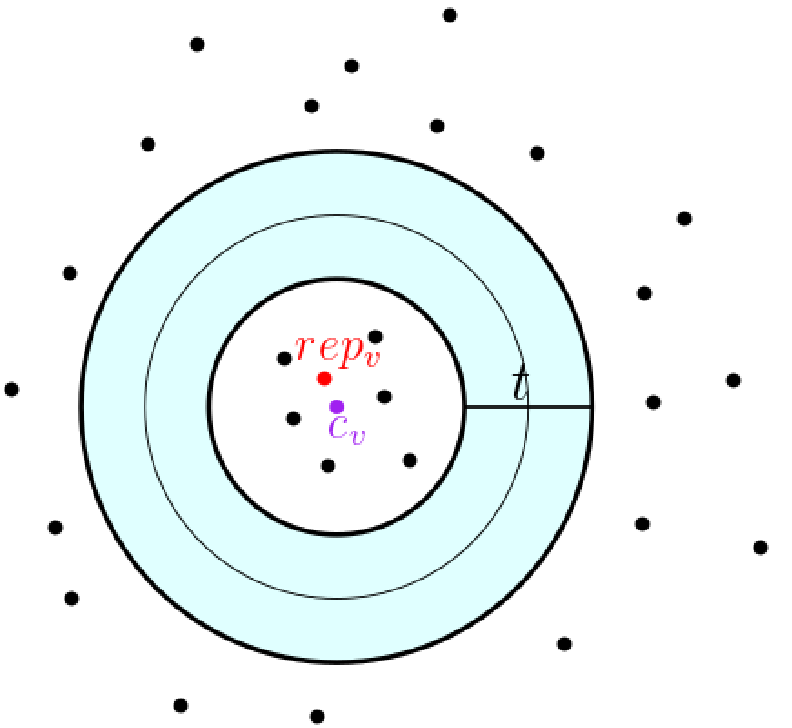
if $||q - c_v|| \leq r_{mid}$ then

$v \leftarrow$ 'inner' child of v

else

$v \leftarrow$ 'outer' child of v

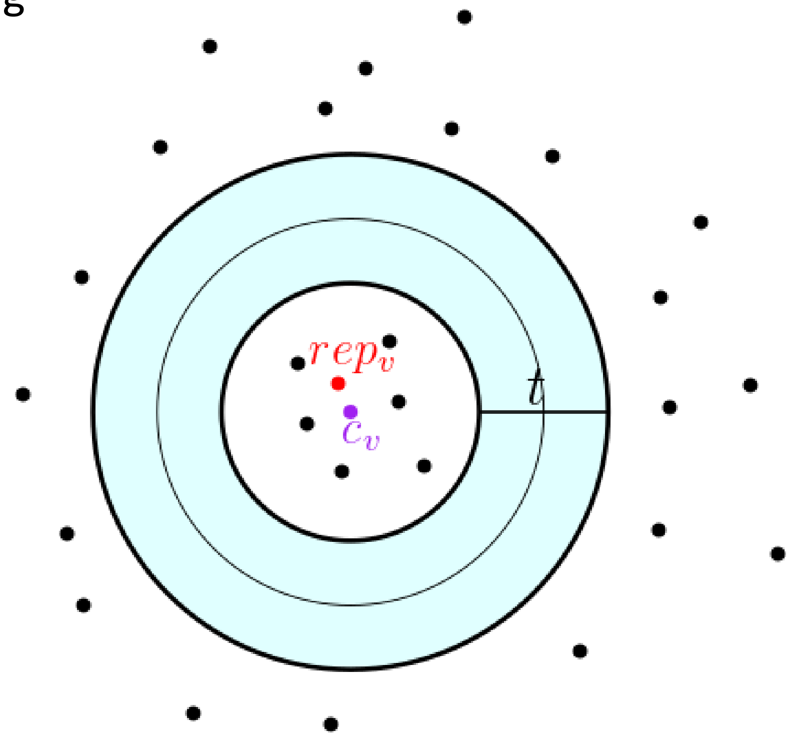
return r_{curr}



Intuition

Case distinction:

- query point is inside the inner ball
- query point is outside the enlarged ball
- query point is in the ring



Intuition

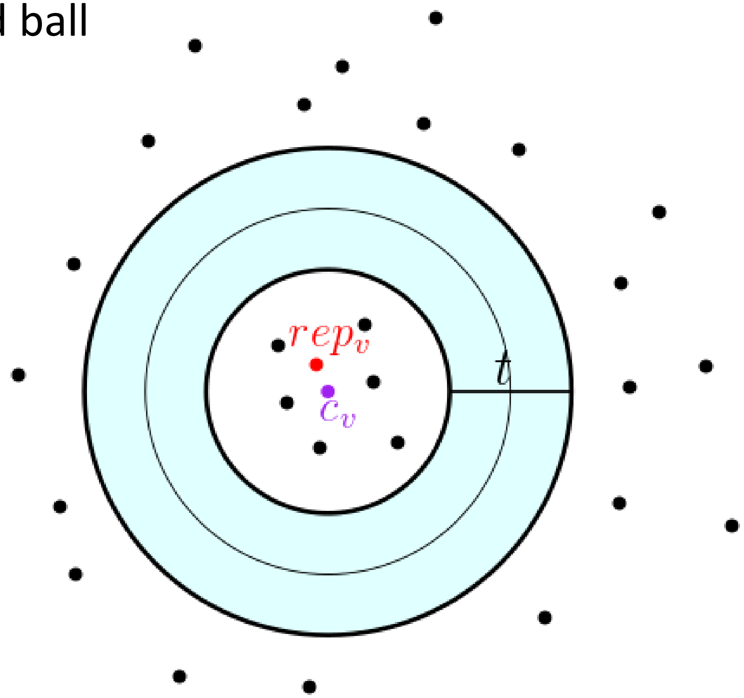
Case distinction:

query point is inside the inner ball

There must be a nearby point, ignore the outside

query point is outside the enlarged ball

query point is in the ring



Intuition

Case distinction:

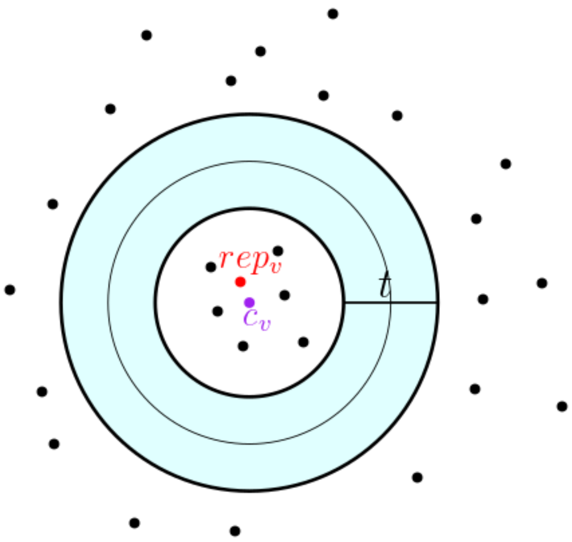
query point is inside the inner ball

There must be a nearby point, ignore the outside

query point is outside the enlarged ball

The inner points are far away, ignore the inside

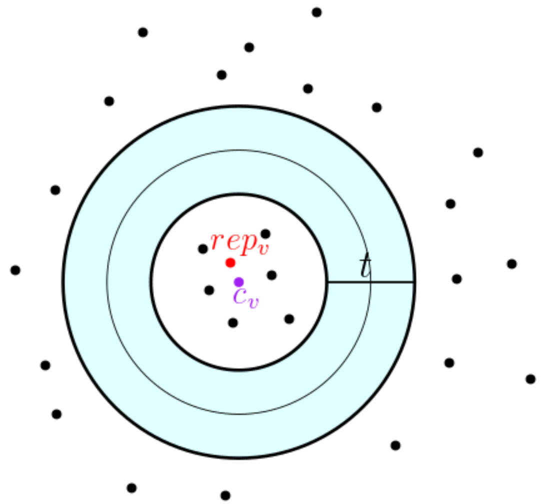
query point is in the ring



Intuition

Case distinction:

- query point is inside the inner ball
  - There must be a nearby point, ignore the outside
- query point is outside the enlarged ball
  - The inner points are far away, ignore the inside
- query point is in the ring
  - Split the ring halfway, group with inner or outer set
  - Slightly worse constants



Proof outline

Let P be a set of n points in R^d. One can preprocess it in O(n log n) time, such that given a query point $q \in R^d$, one can return a (1 + 4n)-ANN of q in P in O(log n) time.

Proof outline

Let P be a set of n points in R^d. One can preprocess it in O(n log n) time, such that given a query point $q \in R^d$, one can return a (1 + 4n)-ANN of q in P in O(log n) time.

Step 1: prove that the approximation factor is met by the search, which takes O(depth(tree)) time

Proof outline

Let P be a set of n points in R^d. One can preprocess it in O(n log n) time, such that given a query point $q \in R^d$, one can return a (1 + 4n)-ANN of q in P in O(log n) time.

Step 1: prove that the approximation factor is met by the search, which takes O(depth(tree)) time

Step 2: prove that the tree of depth O(log n) can be built in O (n log n) time

Proof outline

Let P be a set of n points in R^d. One can preprocess it in O(n log n) time, such that given a query point q ∈ R^d, one can return a (1 + 4n)-ANN of q in P in O(log n) time.

Step 1: prove that the approximation factor is met by the search, which takes O(depth(tree)) time

Step 2: prove that the tree of depth O(log n) can be built in O (n log n) time

Step 3: combine previous steps to reach above goal