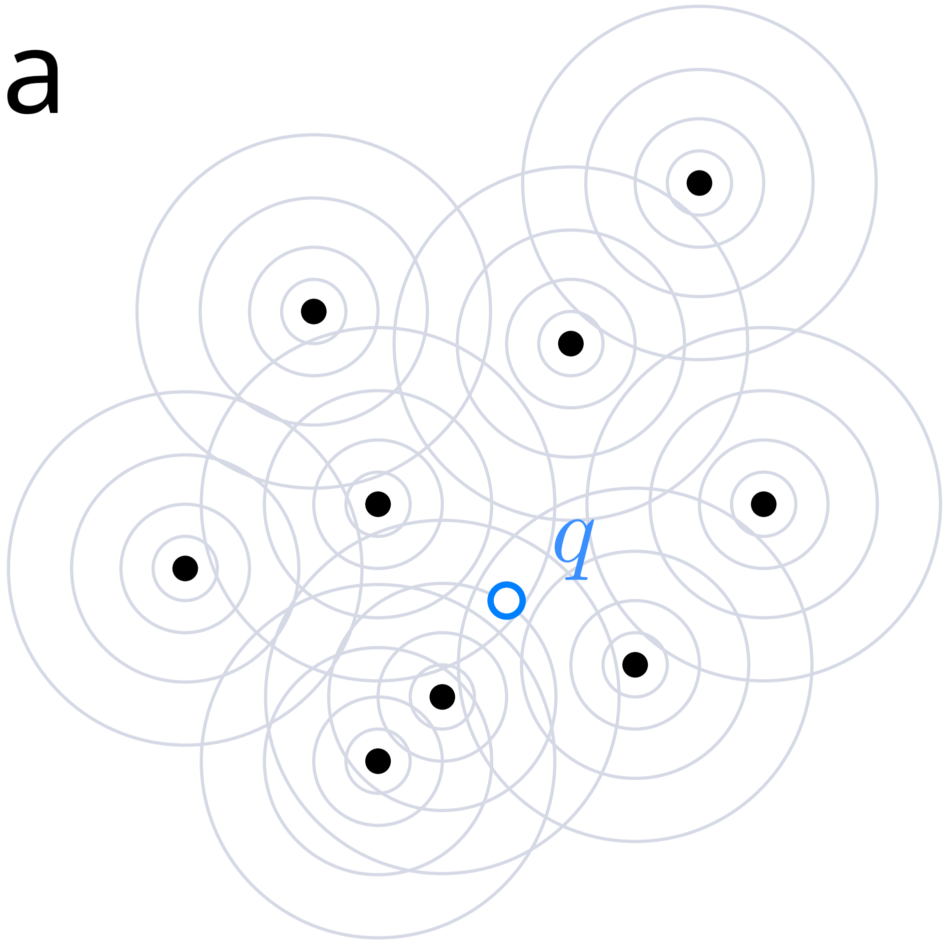
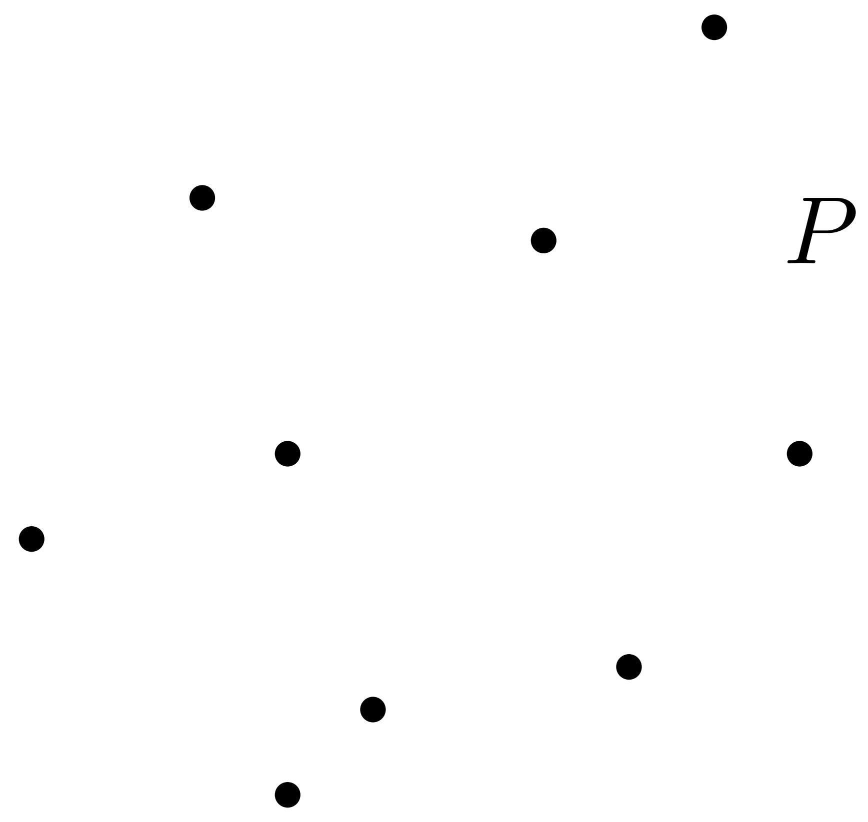
Approximate Nearest Neighbors via Point Location Among Balls



ANN: What happened so far

$(1+\varepsilon)$**- Approximate Nearest Neighbors (ANN)**



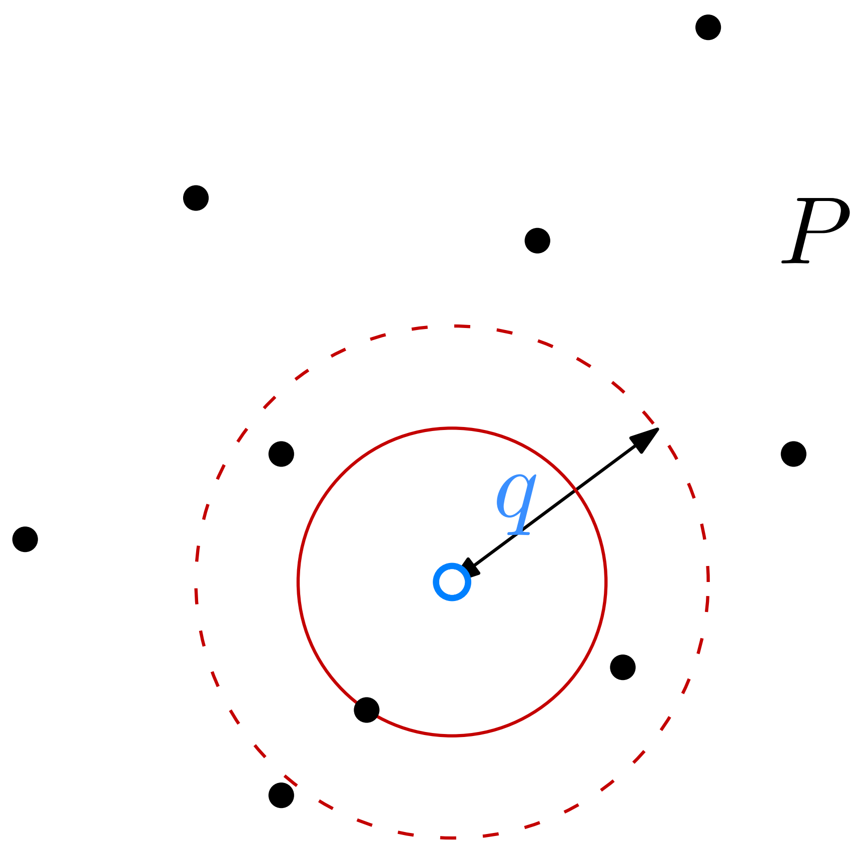
ANN: What happened so far

$(1+\varepsilon)$**- Approximate Nearest Neighbors (ANN)**

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



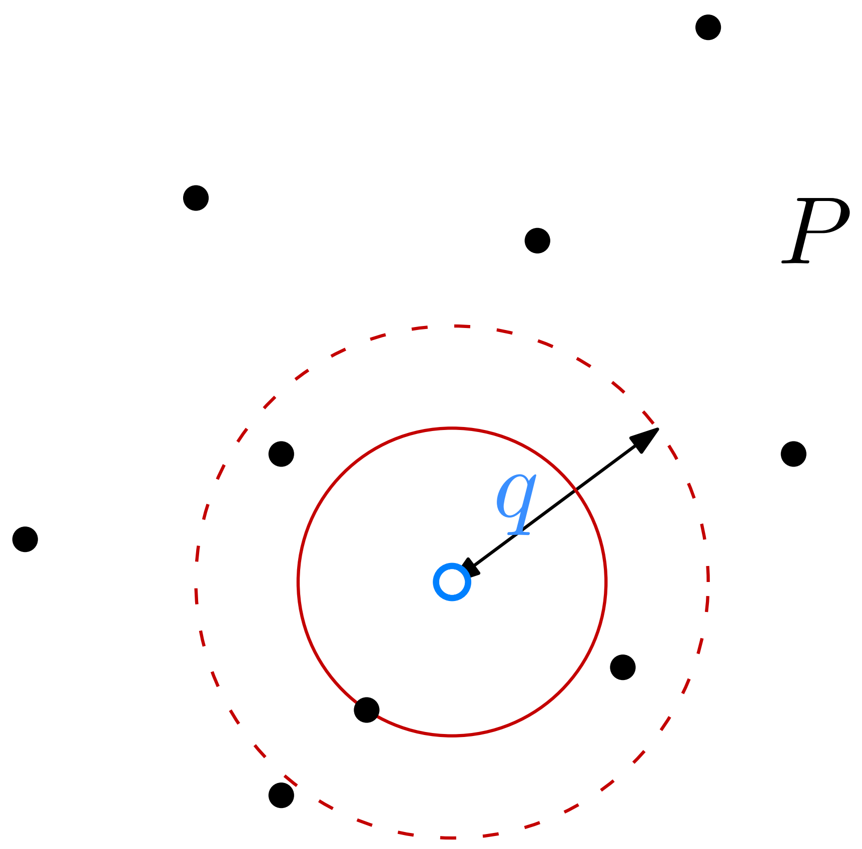
ANN: What happened so far

$(1+\varepsilon)$- Approximate Nearest Neighbors (ANN)

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



Solutions seen so far

?

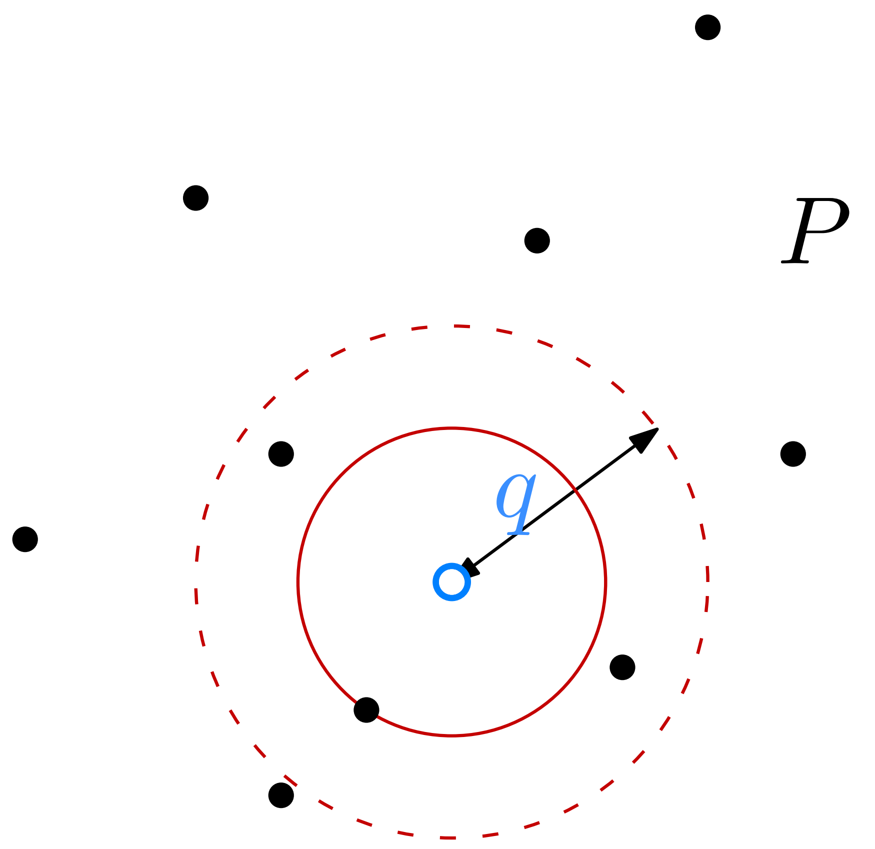
ANN: What happened so far

$(1+\varepsilon)$- Approximate Nearest Neighbors (ANN)

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



Solutions seen so far

quadtrees: $(1+\varepsilon)$-ANN for bounded spread

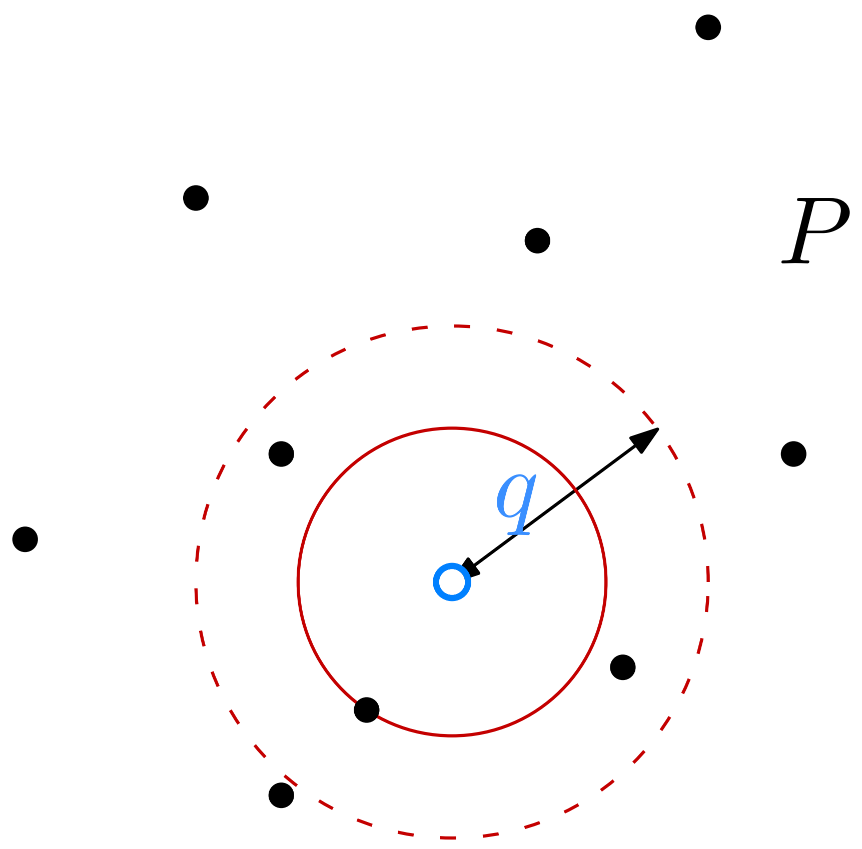
ANN: What happened so far

$(1+\varepsilon)$- Approximate Nearest Neighbors (ANN)

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



Solutions seen so far

quadtrees: $(1+\varepsilon)$-ANN for bounded spread

ring-separator tree: $O(n)$-ANN

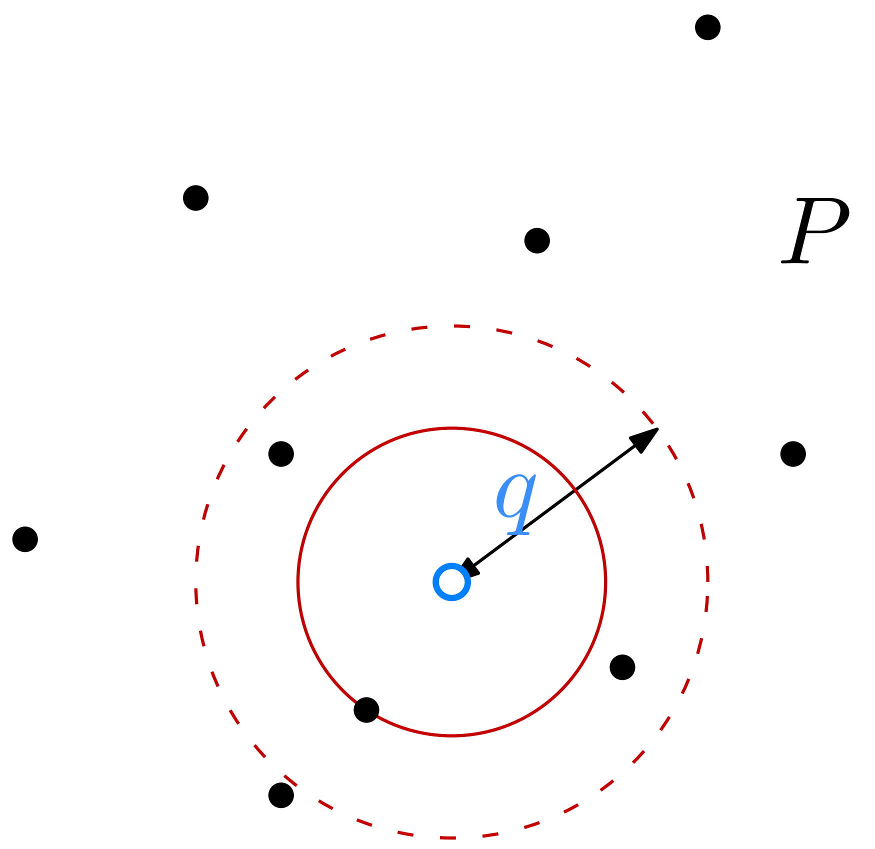
ANN: What happened so far

$(1+\varepsilon)$- Approximate Nearest Neighbors (ANN)

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



Solutions seen so far

quadtrees: $(1+\varepsilon)$-ANN for bounded spread

ring-separator tree: $O(n)$-ANN

shifted quadtrees: $n^{O(1)}$-ANN

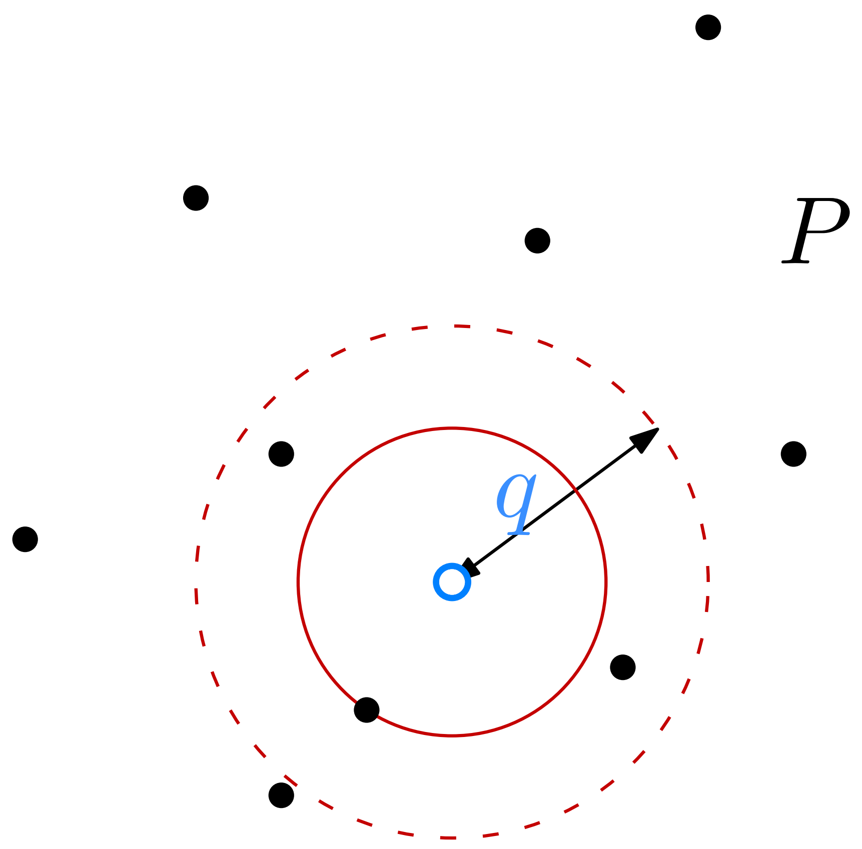
ANN: What happened so far

$(1+\varepsilon)$- Approximate Nearest Neighbors (ANN)

Problem statement:

Given q, P, find p such that $d(q,p) \leq (1+\varepsilon)d(q,P)$

where $d(q,P)$ denotes the smallest distance from q to any other point in P, i.e. distance to its nn



Solutions seen so far

quadtrees: $(1+\varepsilon)$-ANN for bounded spread

ring-separator tree: $O(n)$-ANN

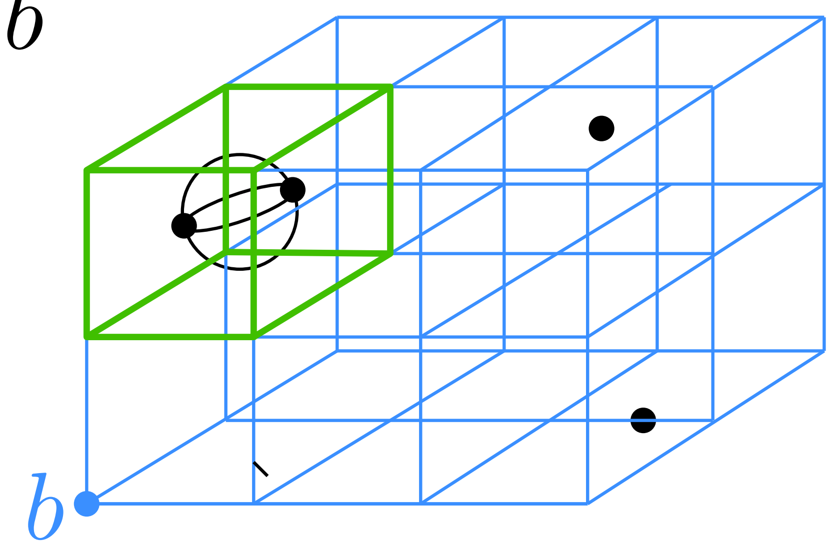
shifted quadtrees: $n^{O(1)}$-ANN

combined: $(1+\varepsilon)$-ANN in low dimensions

ANN: What happened so far

recap: shifting grids and quadtrees by a random vector b

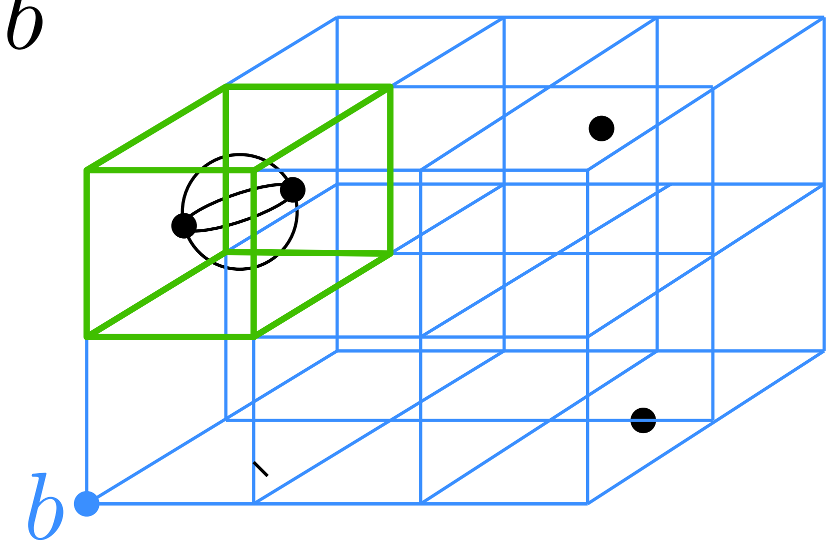
For a ball B of radius r: the probability that B is not in a single cell of $G^d(b, \Delta)$ is at most $\min\left(\frac{2dr}{\Delta}, 1\right)$.



ANN: What happened so far

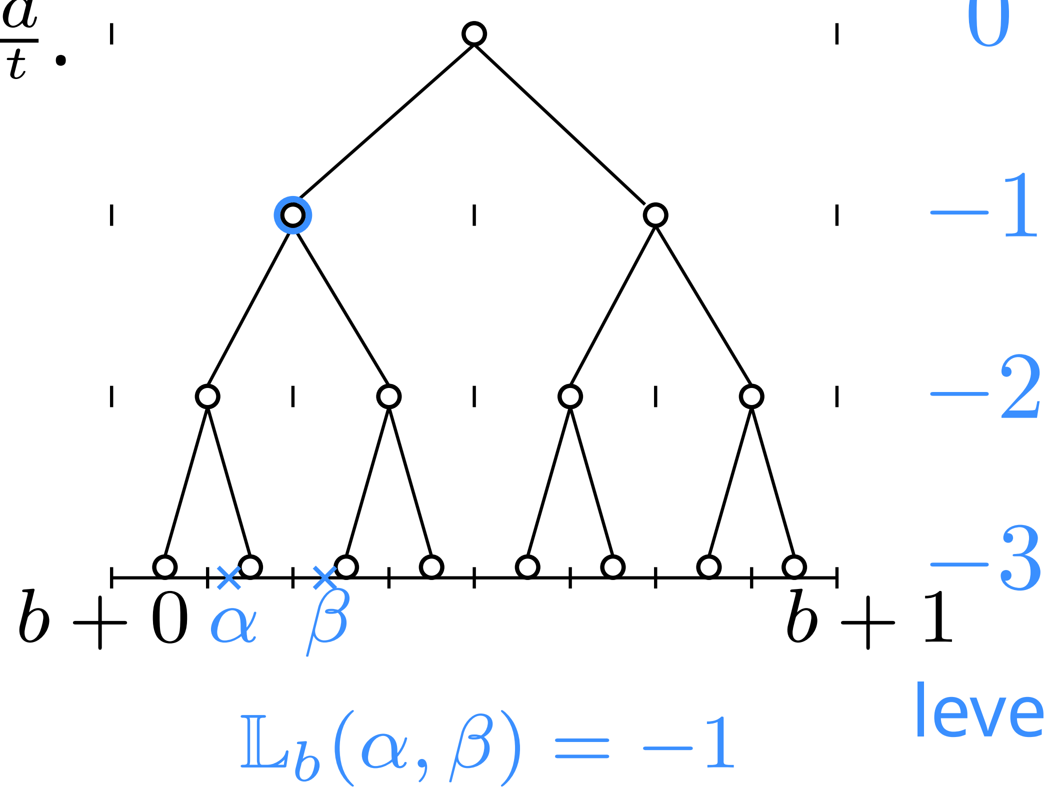
recap: shifting grids and quadtrees by a random vector b

For a ball B of radius r: the probability that B is not in a single cell of $G^d(b, \Delta)$ is at most $\min\left(\frac{2dr}{\Delta}, 1\right)$.



For $t > 0$ holds $\mathbb{P}\left[\mathbb{L}_b(p,q) > \log_2 ||p-q|| + t\right] \le \frac{4d}{2^t}$.

With high probability p and q in same cell at level $\log_2 ||p-q|| + c \log n$ (size of cell: $||p-q|| n^c$)



ANN: today

Approximating a metric space by a hierarchical well-separated tree (HST)

hierachical well-separated trees

simple $(n-1)$-approximation

fast $n^{O(1)}$-approximation in $\mathbb{R}^d$

ANN via point location among balls

simple construction

handling a range of radii

ANN data structure based on HST

ANN: today

Approximating a metric space by a hierarchical well-separated tree (HST)

hierachical well-separated trees

simple $(n-1)$-approximation

fast $n^{O(1)}$-approximation in $\mathbb{R}^d$

ANN via point location among balls

simple construction

handling a range of radii

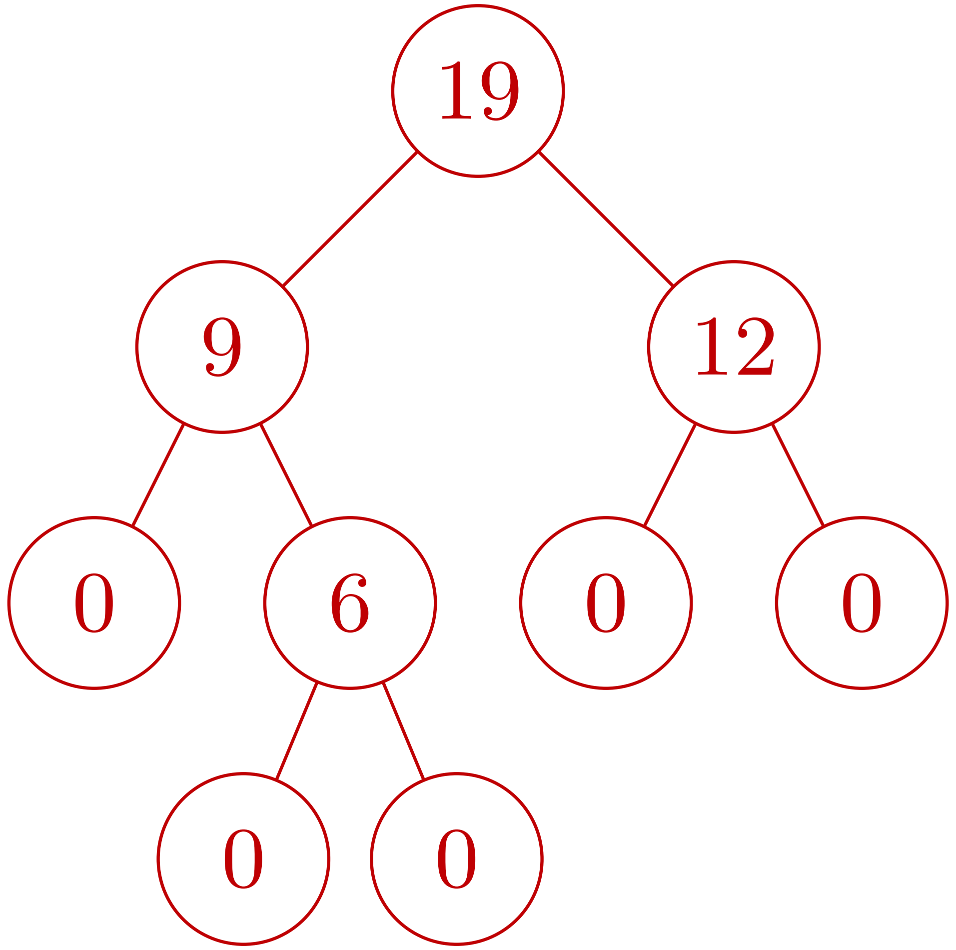
ANN data structure based on HST

. . . **and next time**

point location among approximate balls

approximate Voronoi diagrams

Approximating a metric space by a hierarchical well-separated tree (HST)



Metric space

metric space $M = (X, d)$:

a set X

a distance function $d\colon X \times X \to [0, \infty)$

?

Metric space

metric space $M = (X, d)$:

a set X

a distance function $d\colon X \times X \to [0, \infty)$

- $d(x, y) = 0$ iff x=y
- $d(x, y) = d(y, x)$
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)

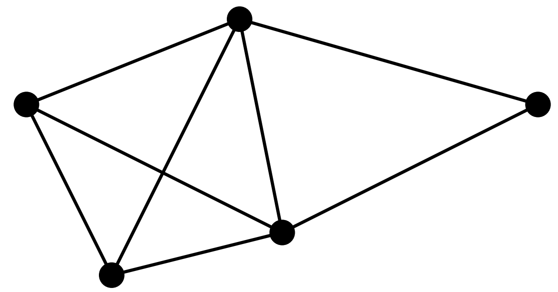
Metric space

metric space $M = (X, d)$:

a set X

a distance function $d\colon X \times X \to [0, \infty)$

- $d(x, y) = 0$ iff x=y
- $d(x, y) = d(y, x)$
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)



For n "points" $P \subset X$:

metric can be represented as matrix of size $\Theta(n^2)$

metric can be represented as weighted graph G with $d(x, y) = dist_G(x, y)$

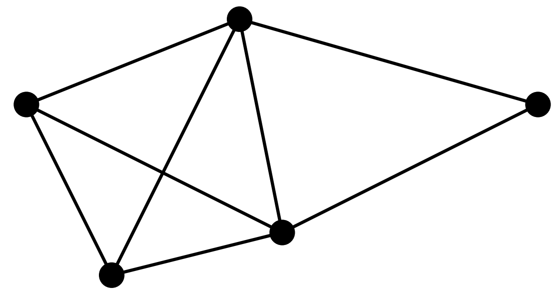
Metric space

metric space $M = (X, d)$:

a set X

a distance function $d\colon X \times X \to [0, \infty)$

- $d(x, y) = 0$ iff x=y
- $d(x, y) = d(y, x)$
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)



For n “points” $P \subset X$:

metric can be represented as matrix of size $\Theta(n^2)$

metric can be represented as weighted graph G with $d(x, y) = dist_G(x, y)$

Question: Examples of metric spaces?

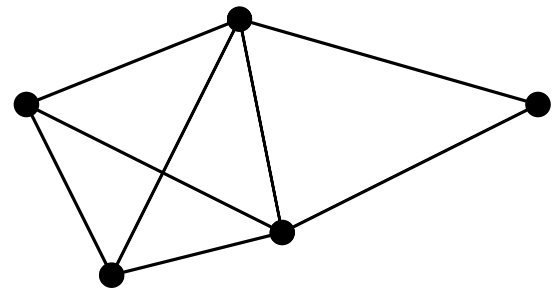
Metric space

metric space $M = (X, d)$:

a set X

a distance function $d\colon X \times X \to [0, \infty)$

- $d(x, y) = 0$ iff x=y
- $d(x, y) = d(y, x)$
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)



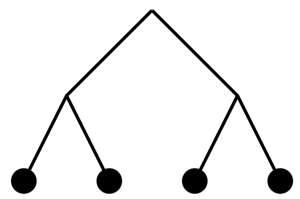
For n “points” $P \subset X$:

metric can be represented as matrix of size $\Theta(n^2)$

metric can be represented as weighted graph G with $d(x, y) = dist_G(x, y)$

We want:

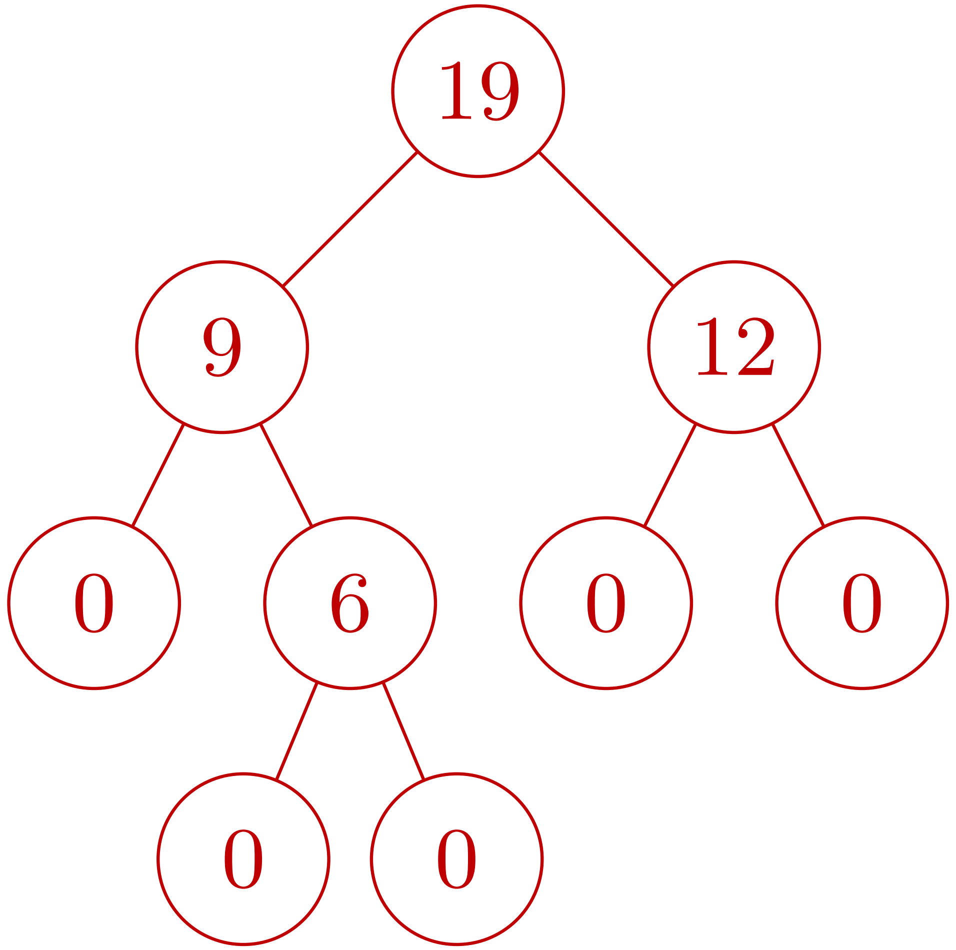
compact, hierarchical(, approximate) representation



Hierarchically well-separated tree (HST)

HST: rooted (binary) tree T over P with

- label $\Delta_v \geq 0$ for each node $v \in T$
- each leaf u_p uniquely corresponds to a point $p \in P$; $\Delta_u = 0$ for all leaves u.
- If u is child of v: $\Delta_u \leq \Delta_v$



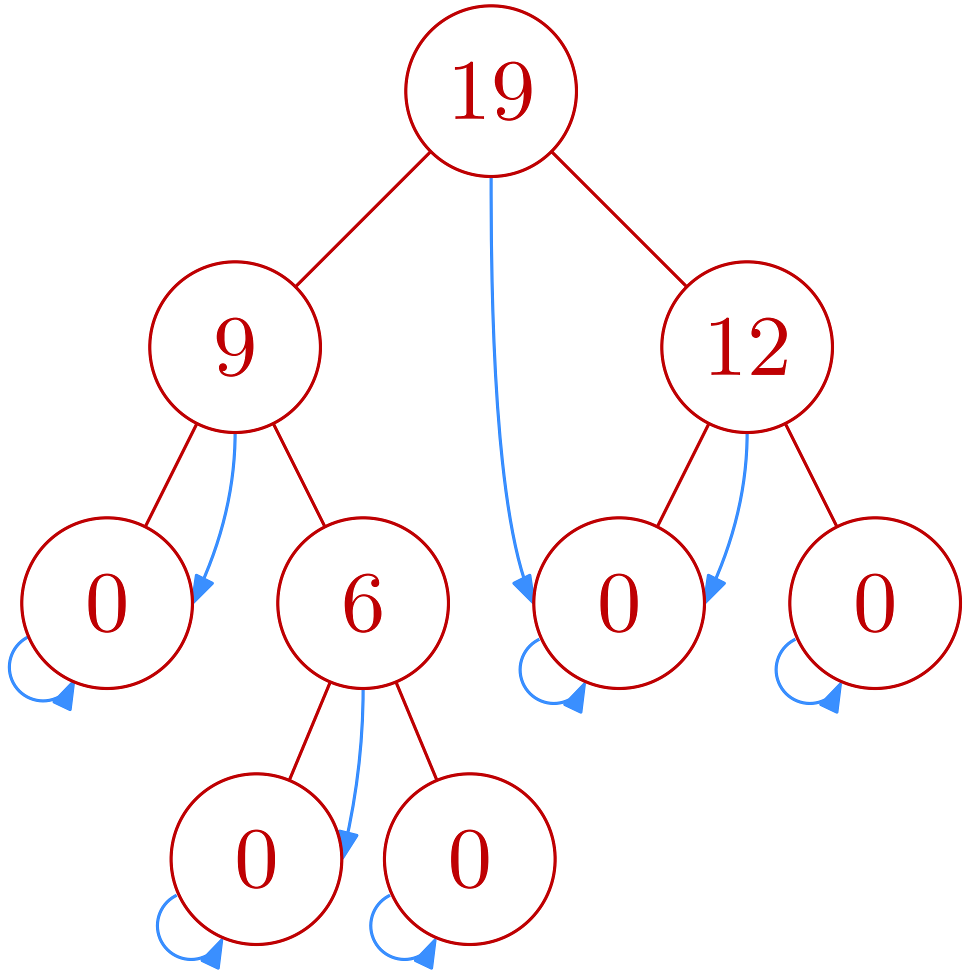
Hierarchically well-separated tree (HST)

HST: rooted (binary) tree T over P with

- label $\Delta_v \geq 0$ for each node $v \in T$
- each leaf u_p uniquely corresponds to a point $p \in P$; $\Delta_u = 0$ for all leaves u.
- If u is child of v: $\Delta_u \leq \Delta_v$

storing representatives:

- for leaf u_p: $\text{repr}_{u_p} = p$
- internal node v: $\text{repr}_v \in \{\text{repr}_v | v \text{ is a child of } u\}$



Hierarchically well-separated tree (HST)

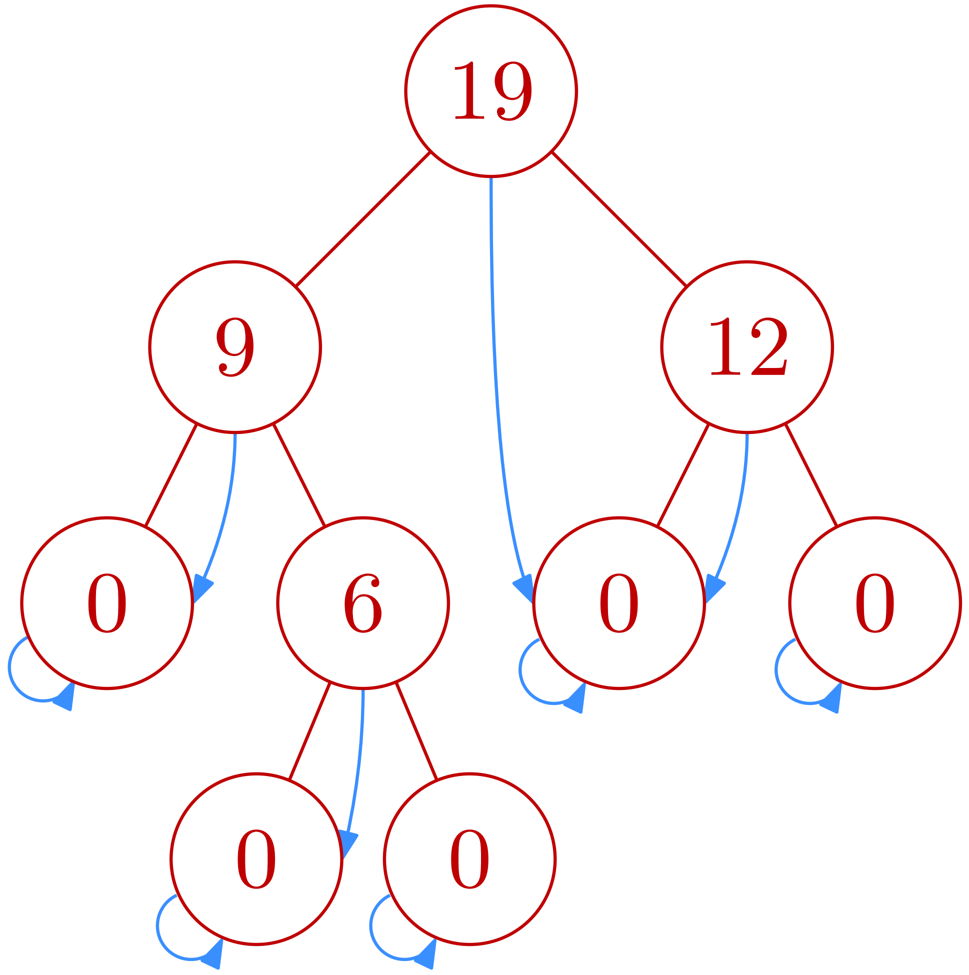
HST: rooted (binary) tree T over P with

- label $\Delta_v \geq 0$ for each node $v \in T$
- each leaf u_p uniquely corresponds to a point $p \in P$; $\Delta_u = 0$ for all leaves u.
- If u is child of v: $\Delta_u \leq \Delta_v$

storing representatives:

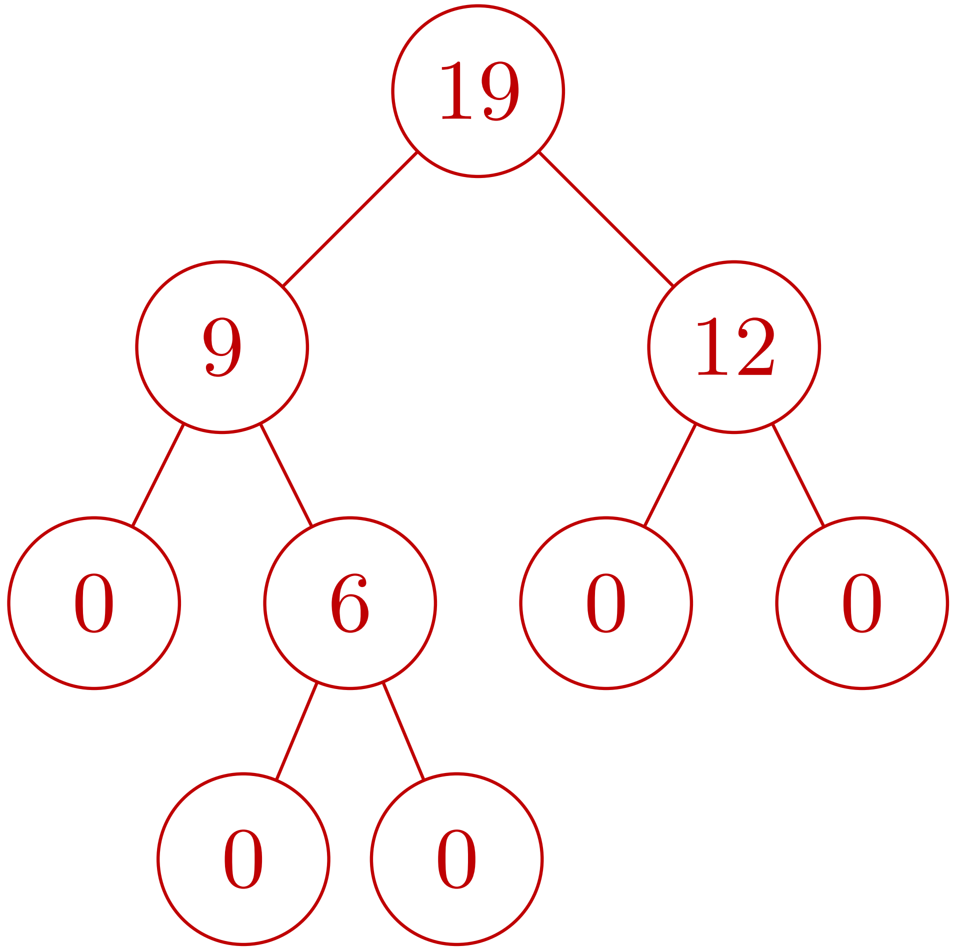
- for leaf u_p: $\text{repr}_{u_p} = p$
- internal node v: $\text{repr}_v \in \{\text{repr}_v | v \text{ is a child of } u\}$

example: quadtree with $\Delta_v =$ diameter of cell



Hierarchically well-separated tree (HST)

$d_T(p, q) := \Delta_{lca(u_p, u_q)}$ is a metric (lca: least common ancestor)

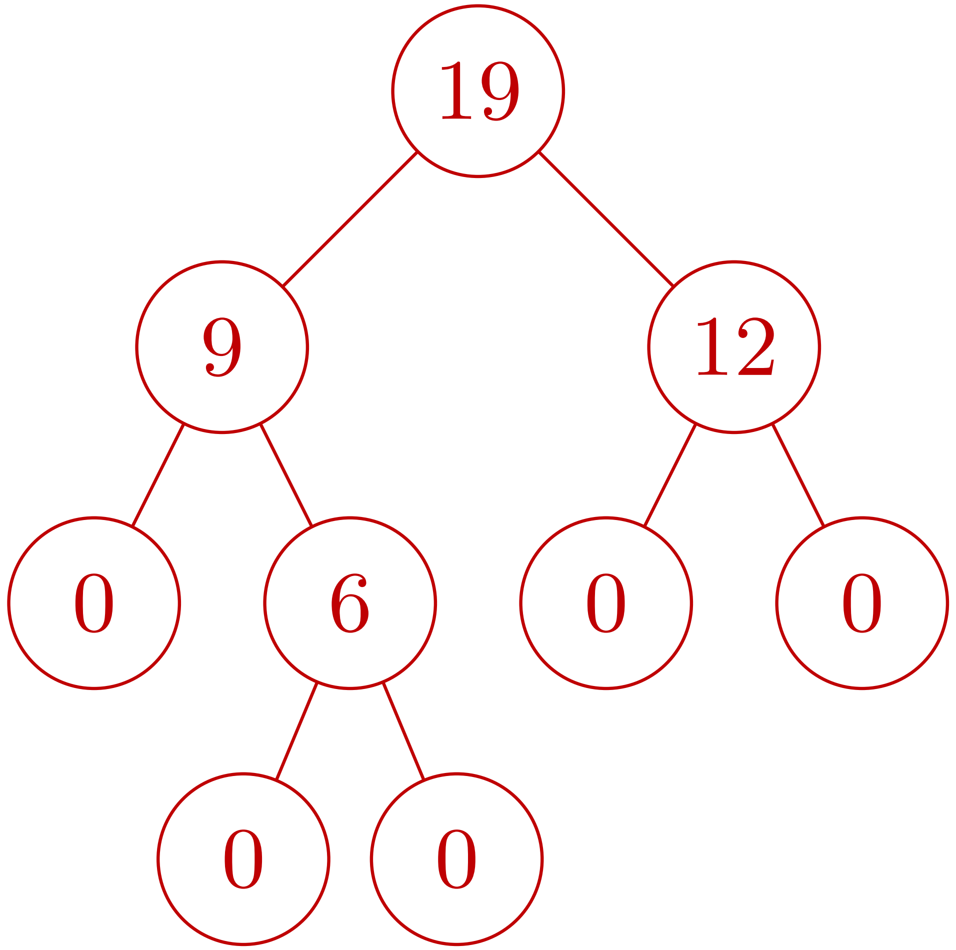


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$

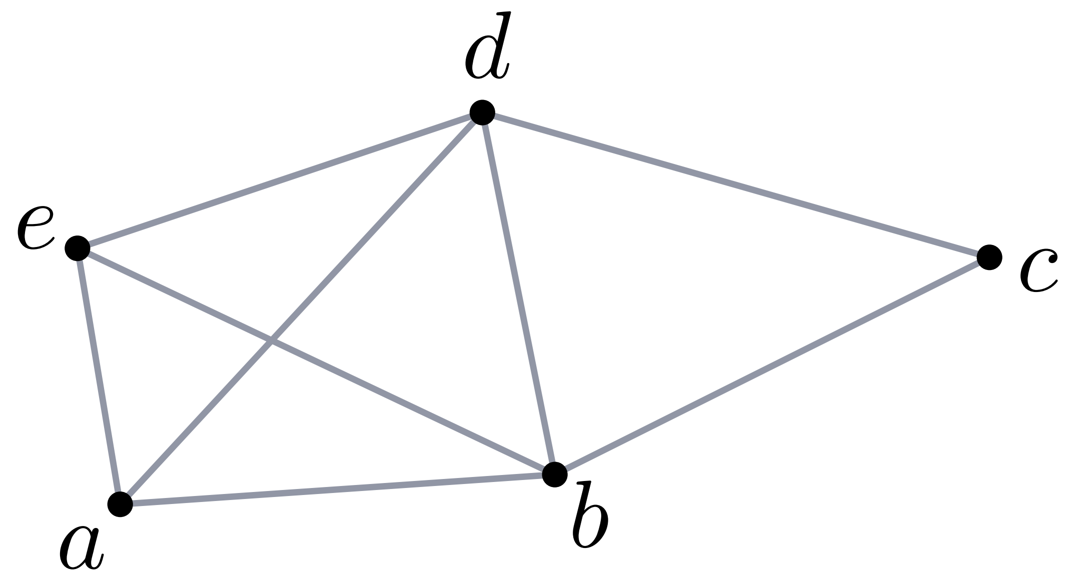


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

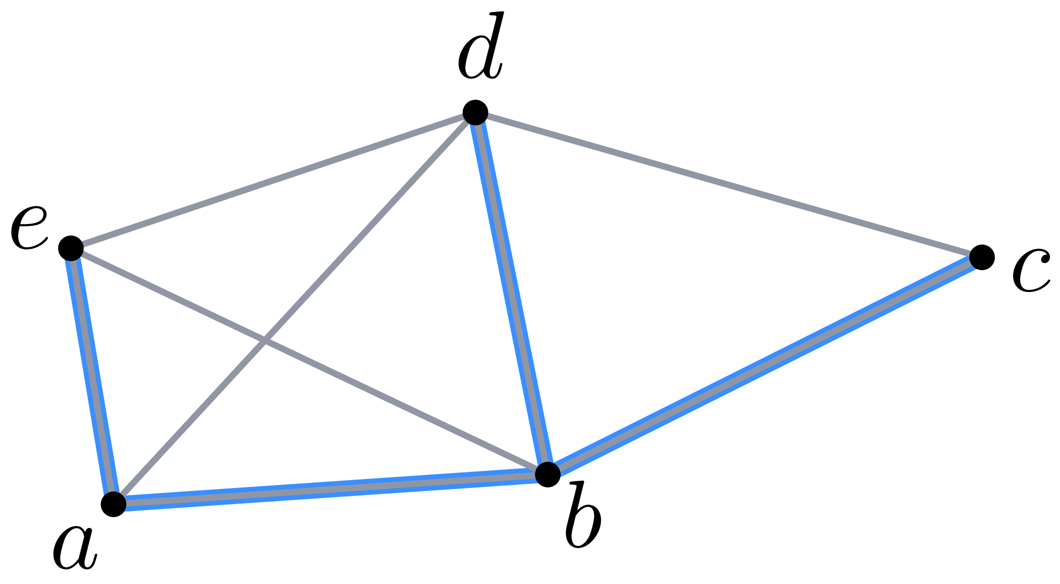
given: metric $M = (P,d)$ as weighted graph

Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

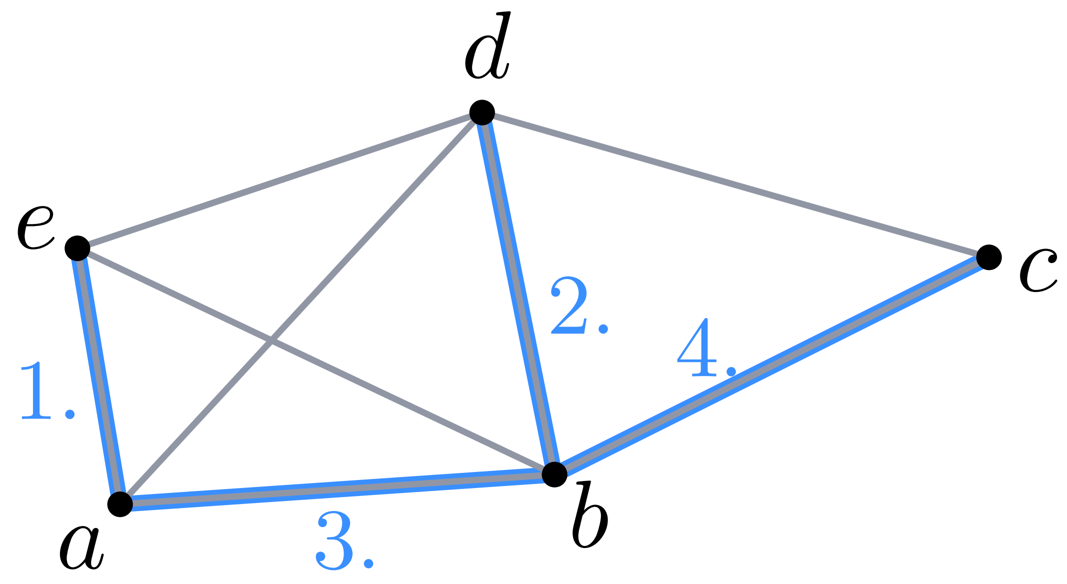
1. compute minimum spanning tree

Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P,d)$ as weighted graph

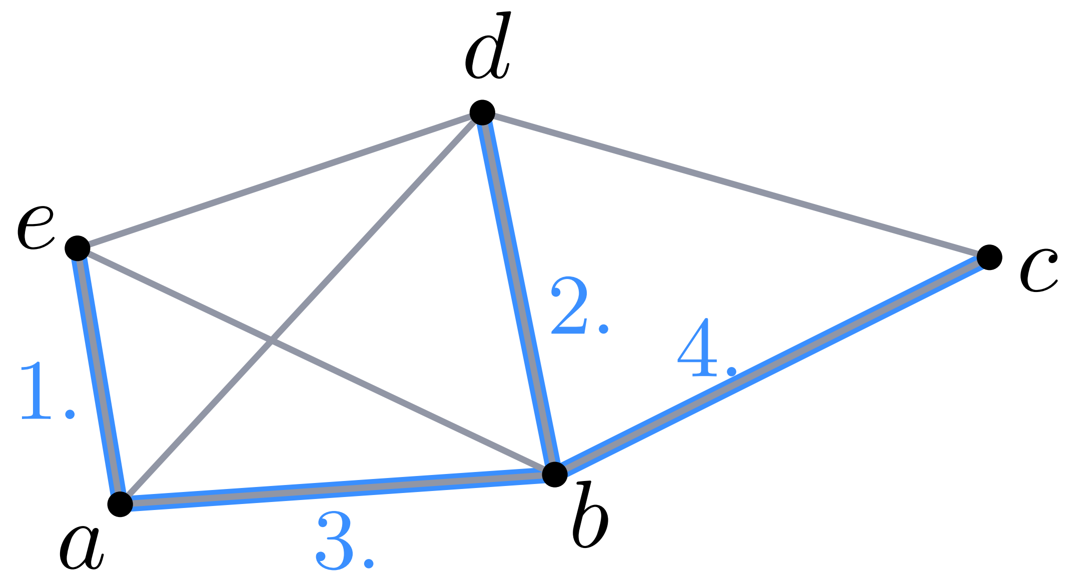
1. compute minimum spanning tree
2. sort edges of MST from short to long

Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

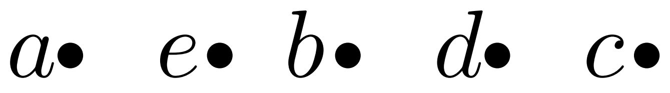
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children

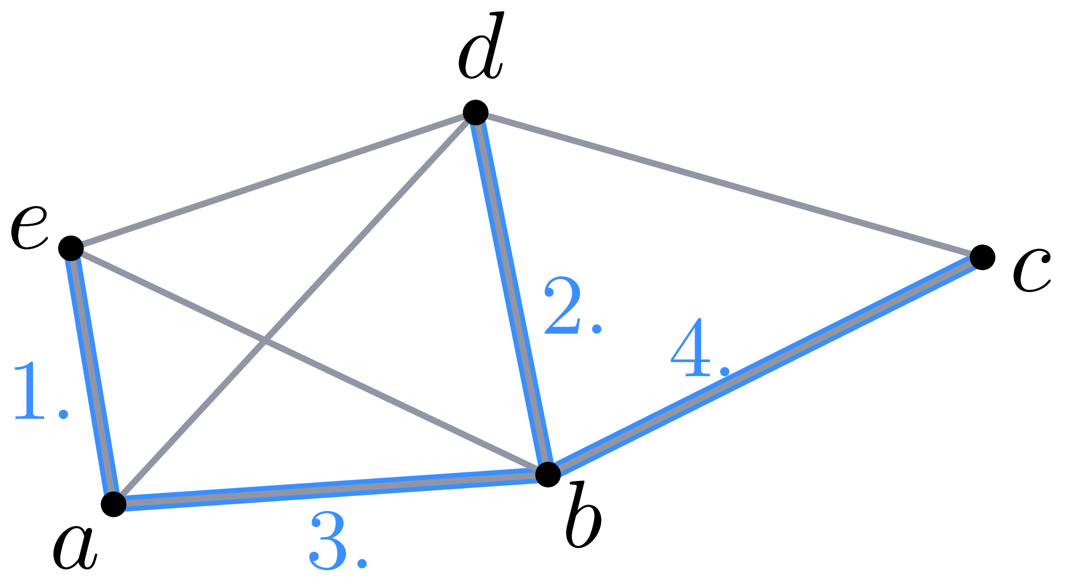


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

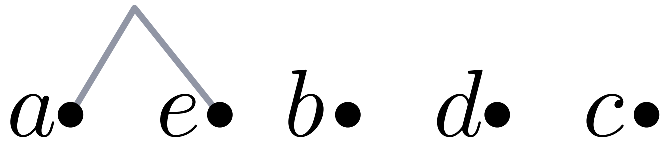
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P,d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children

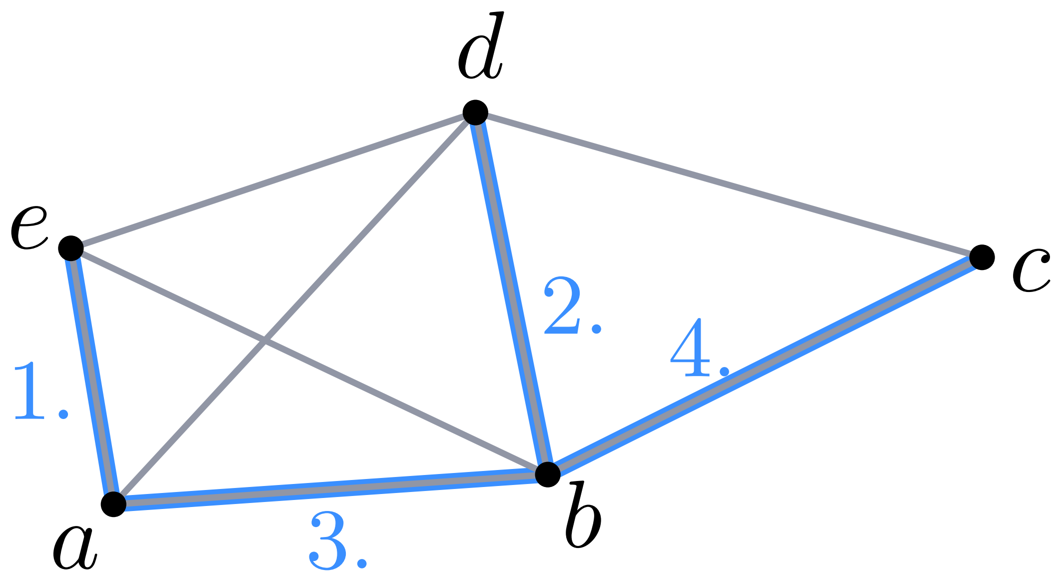


Hierarchically well-separated tree (HST)

$d_T(p, q) := \Delta_{lca(u_p, u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

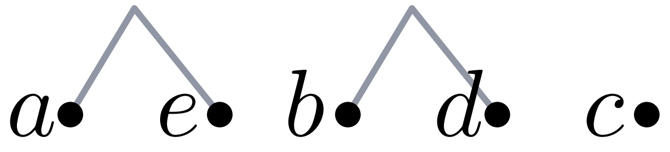
$$d_M(x, y) \leq d_N(x, y) \leq t \cdot d_M(x, y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children

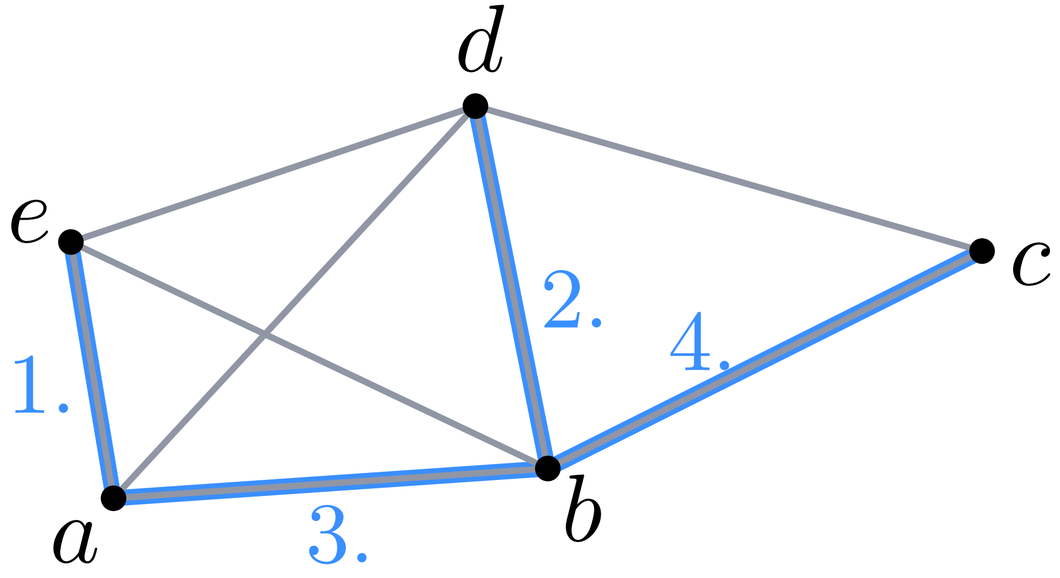


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

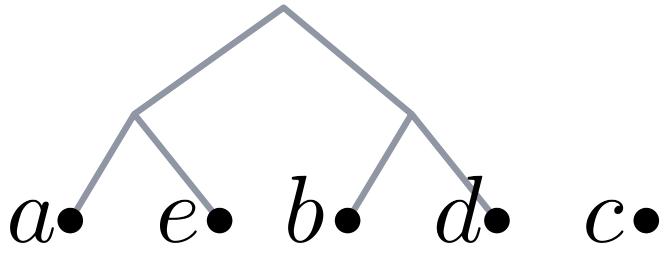
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children

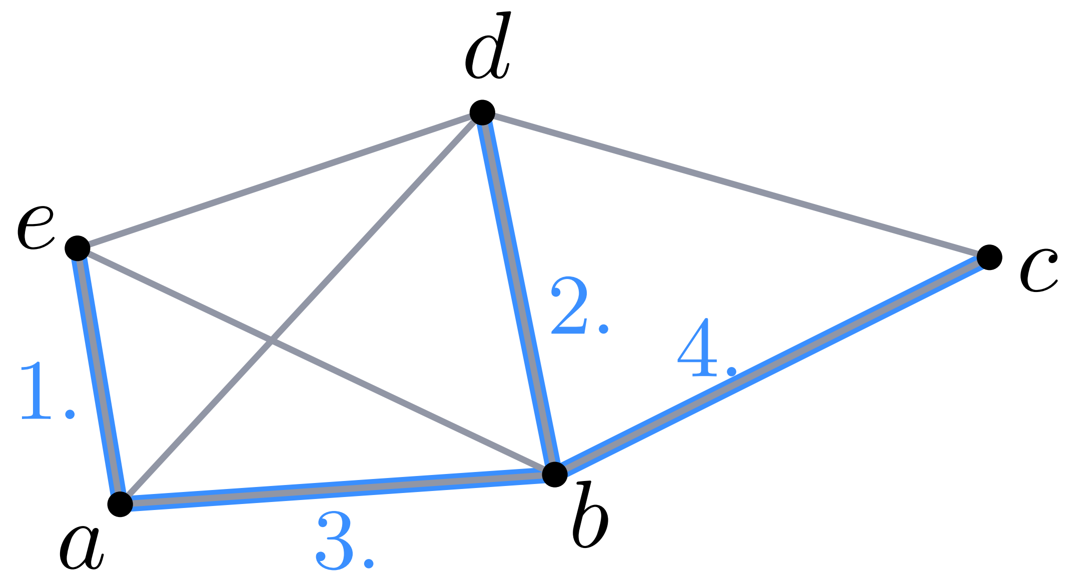


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

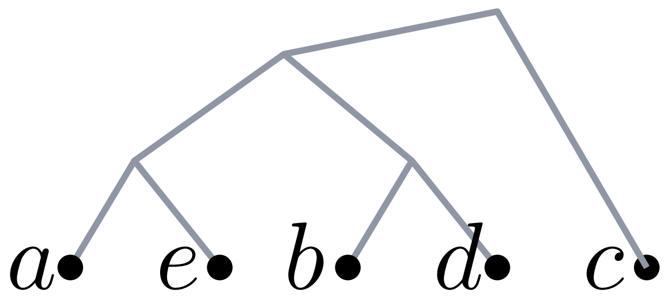
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children

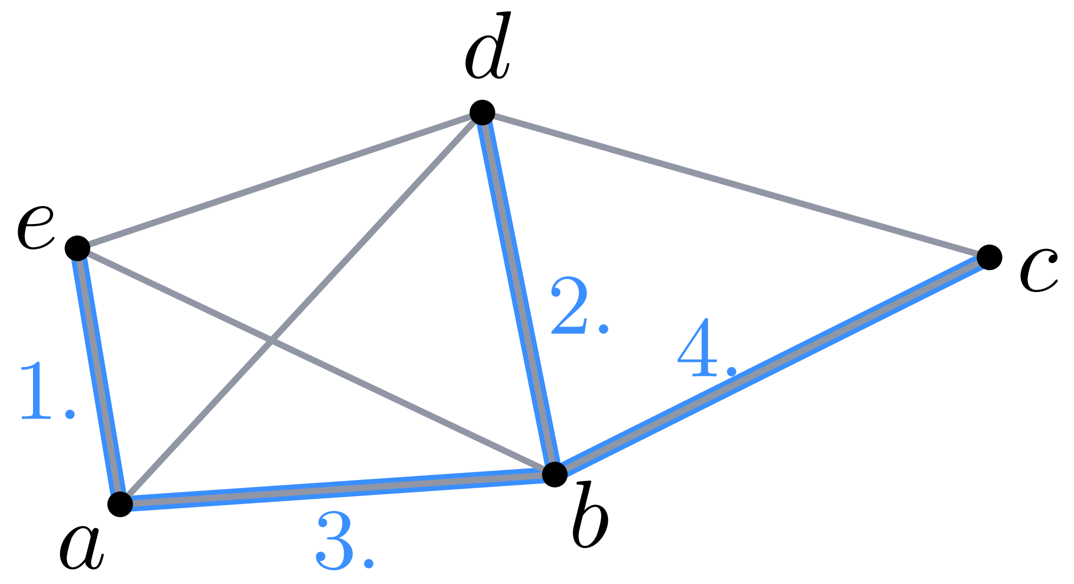


Hierarchically well-separated tree (HST)

$d_T(p, q) := \Delta_{lca(u_p, u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

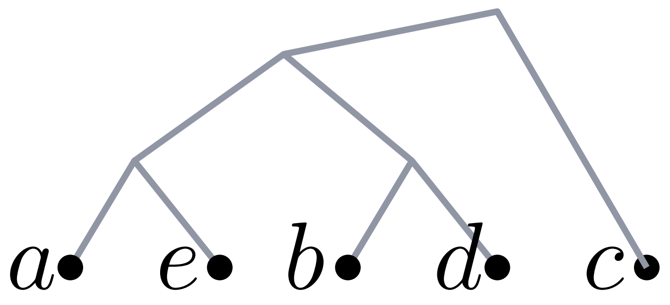
$$d_M(x, y) \leq d_N(x, y) \leq t \cdot d_M(x, y)$$



simple HST construction:

given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children
4. set Δ_v = 0 for leaves
5. for v with point set P_v in subtree and MST edge weight w: $\Delta_v := (|P_v| - 1)\, w$

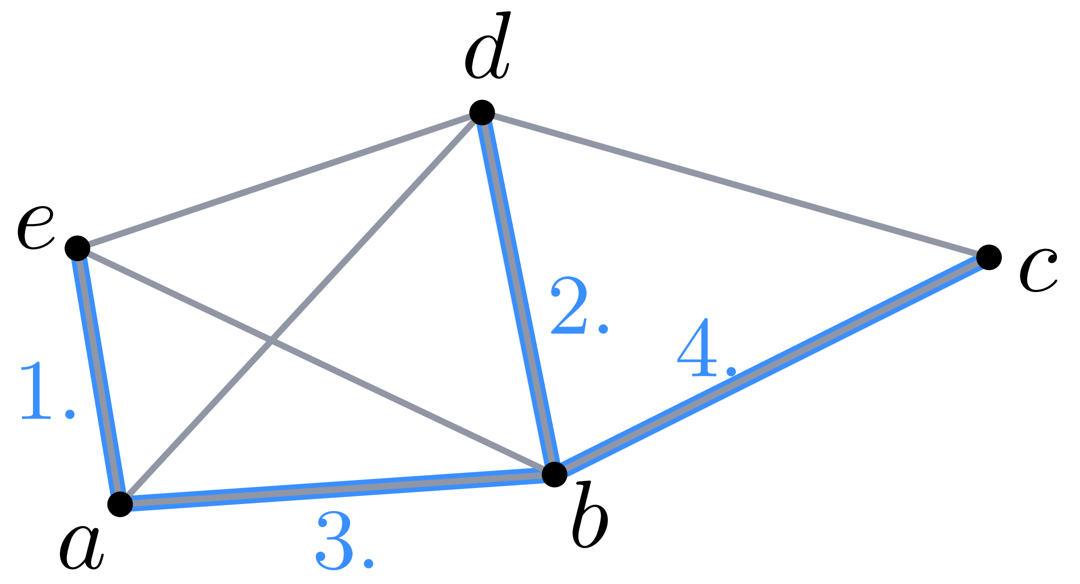


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

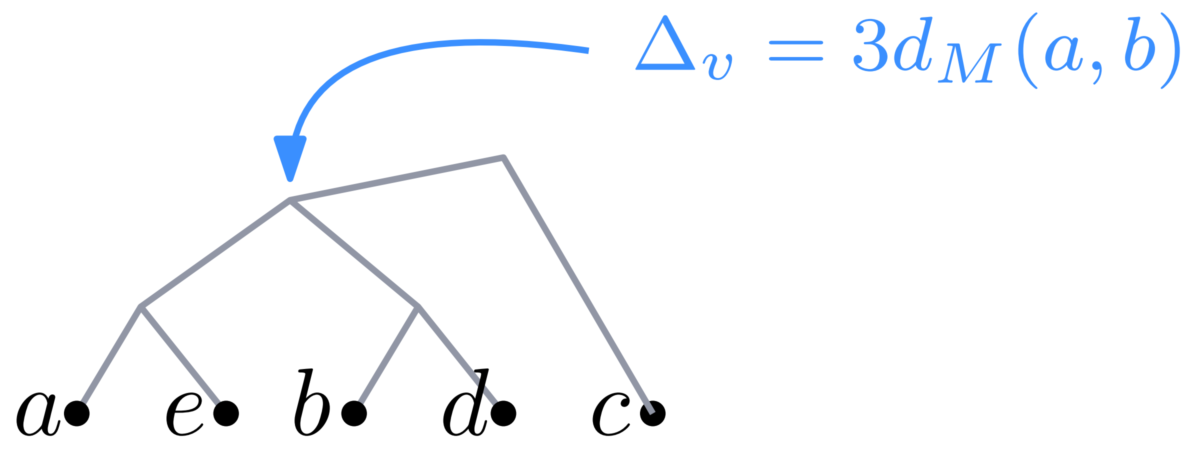
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

given: metric $M = (P,d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children
4. set Δ_v = 0 for leaves
5. for v with point set P_v in subtree and MST edge weight w: $\Delta_v := (|P_v| - 1)\, w$

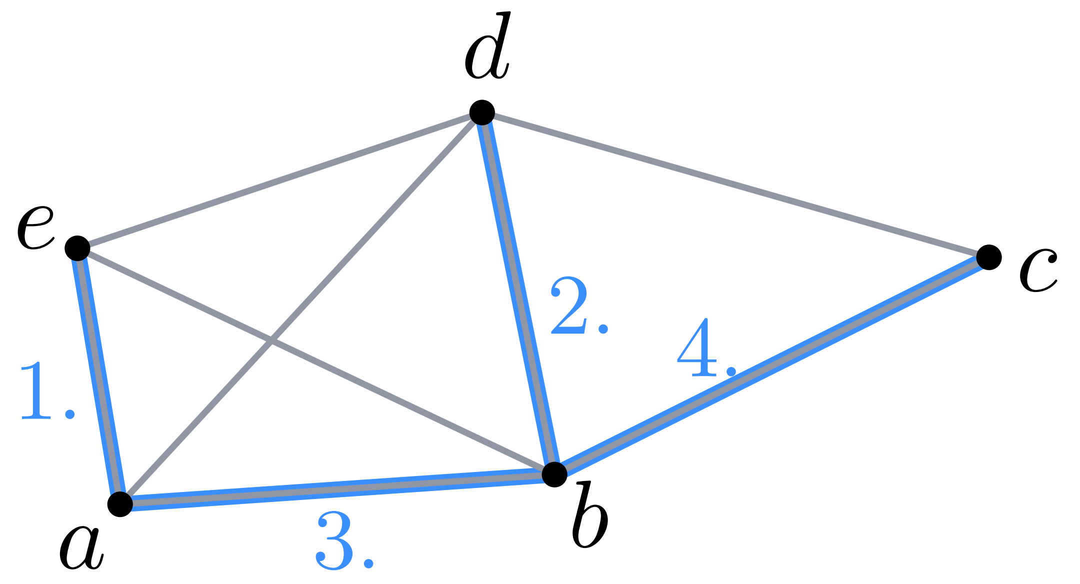


Hierarchically well-separated tree (HST)

$d_T(p,q) := \Delta_{lca(u_p,u_q)}$ is a metric (lca: least common ancestor)

metric N t-approximates metric M:

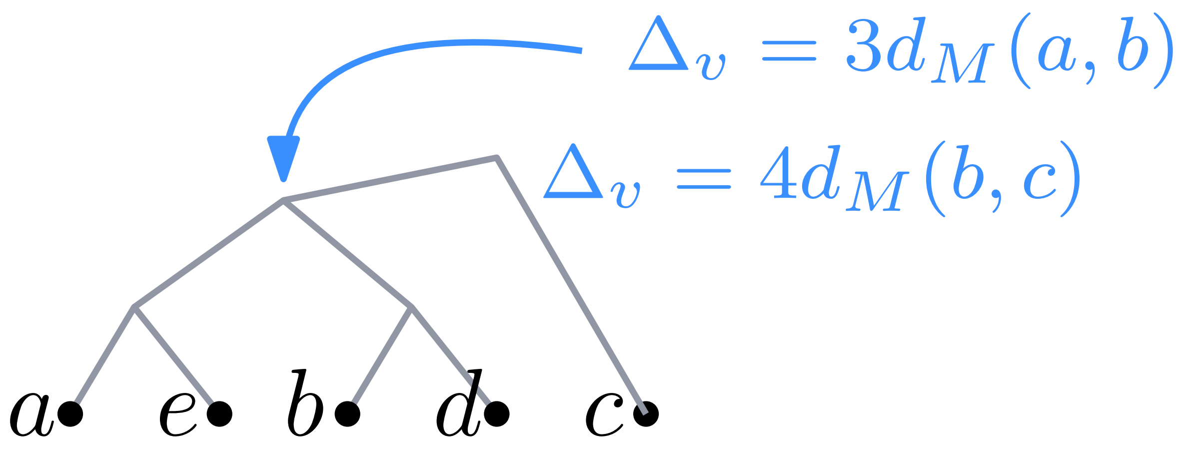
$$d_M(x,y) \leq d_N(x,y) \leq t \cdot d_M(x,y)$$



simple HST construction:

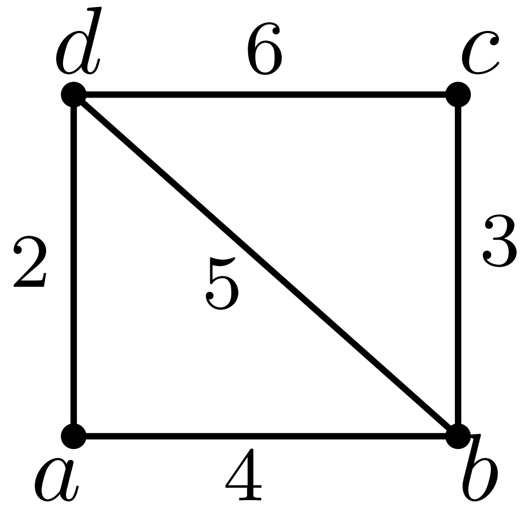
given: metric $M = (P, d)$ as weighted graph

1. compute minimum spanning tree
2. sort edges of MST from short to long
3. build HST bottom-up: add vertex for MST edge with the two merged components as children
4. set Δ_v = 0 for leaves
5. for v with point set P_v in subtree and MST edge weight w: $\Delta_v := (|P_v| - 1)\, w$



Small Assignment

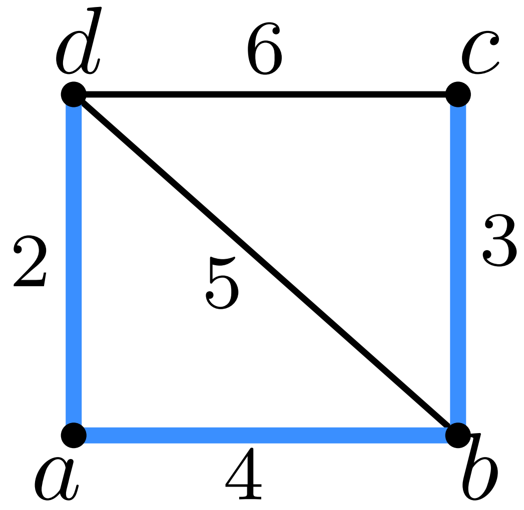
Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?



$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

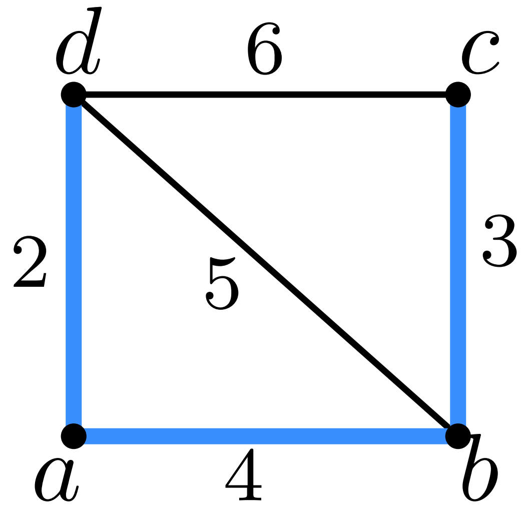
Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

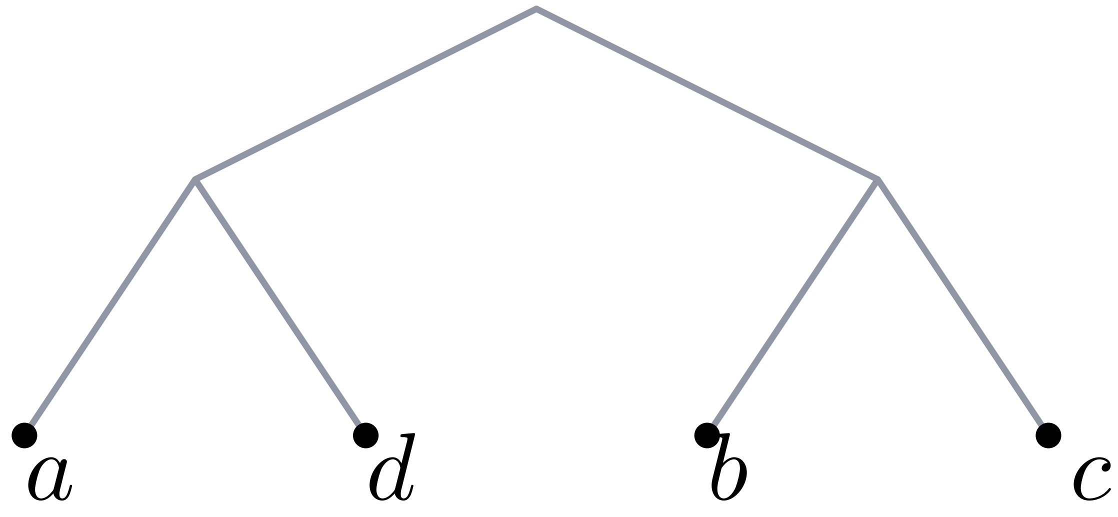


$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

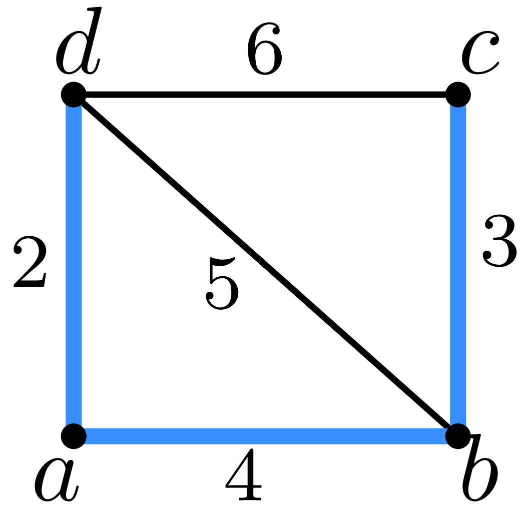


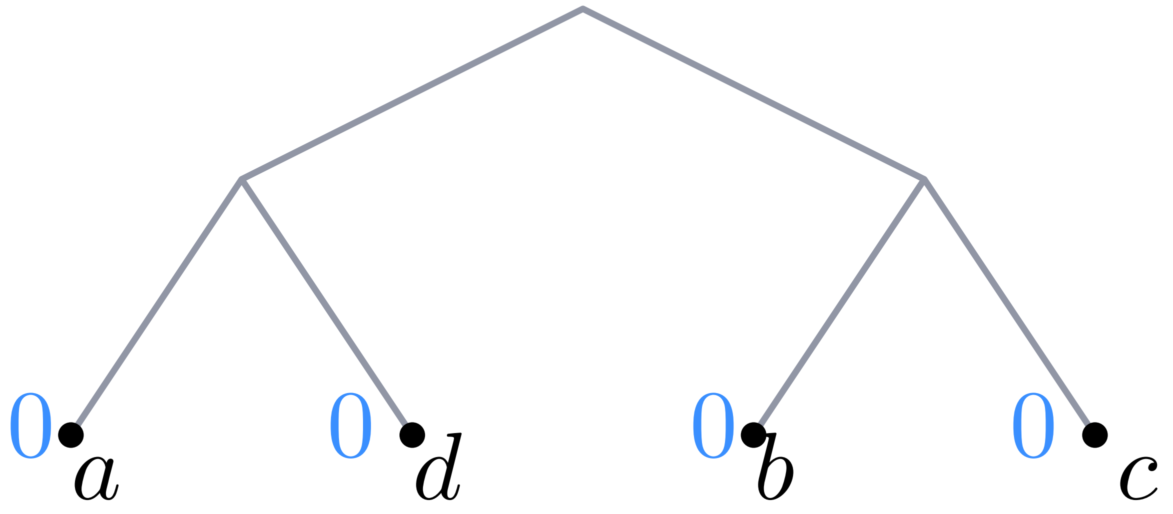


$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

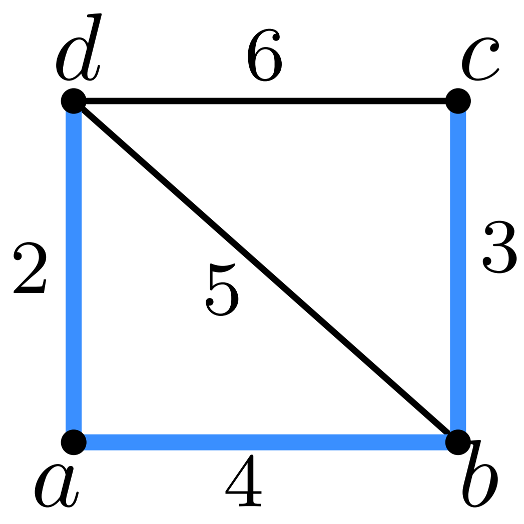


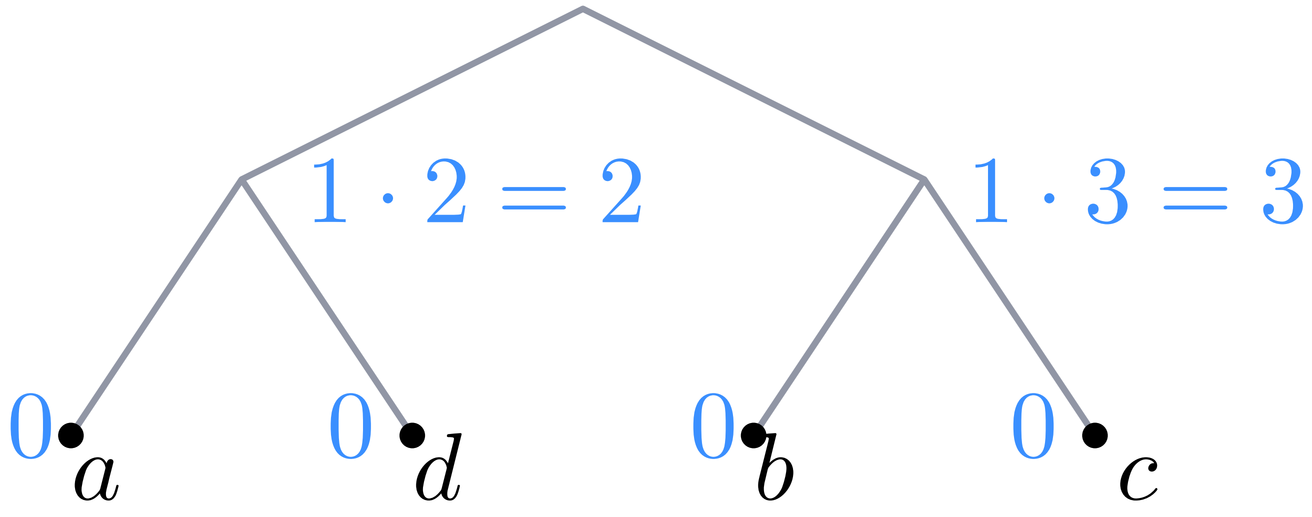


$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

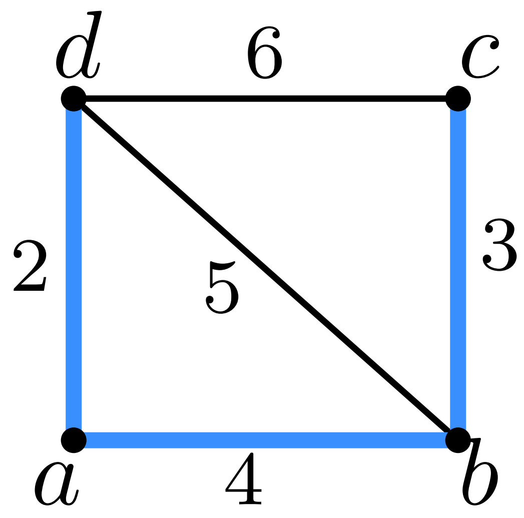


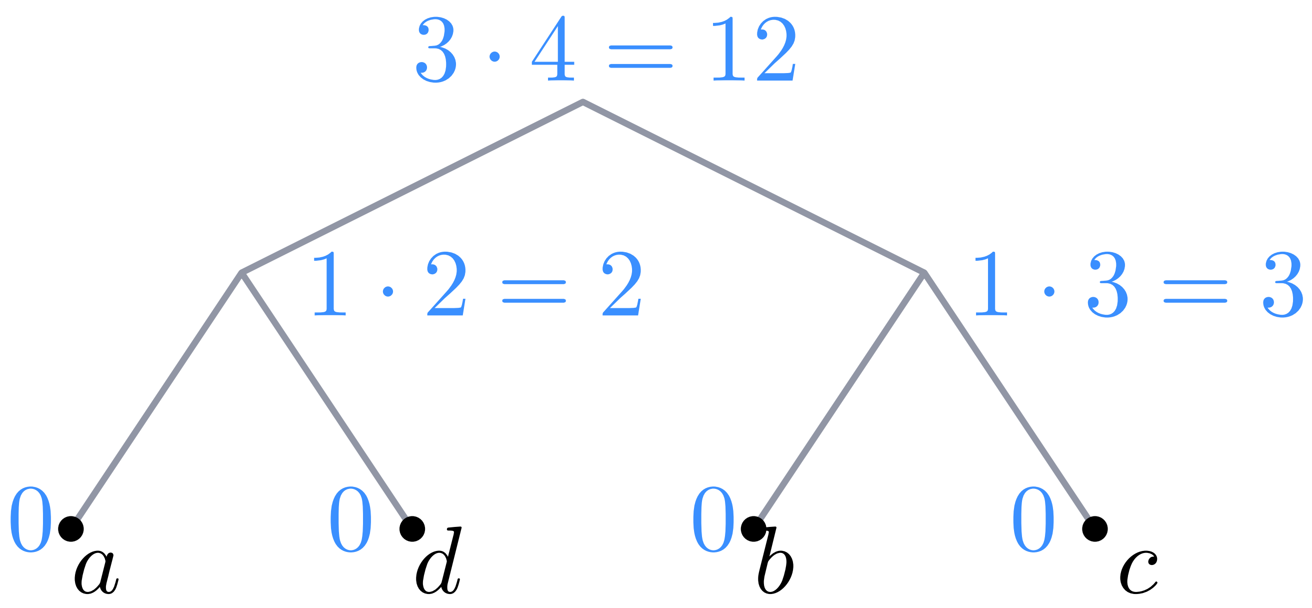


$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

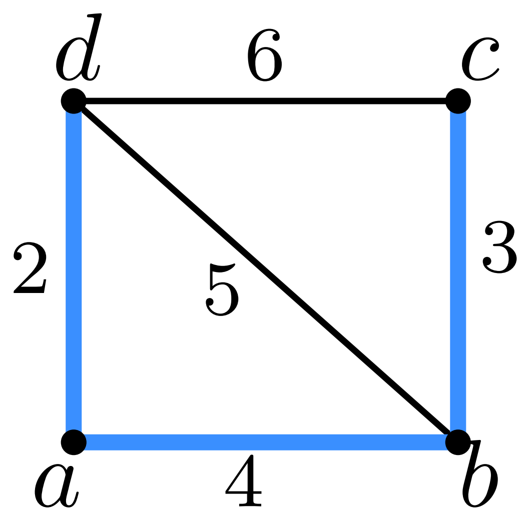




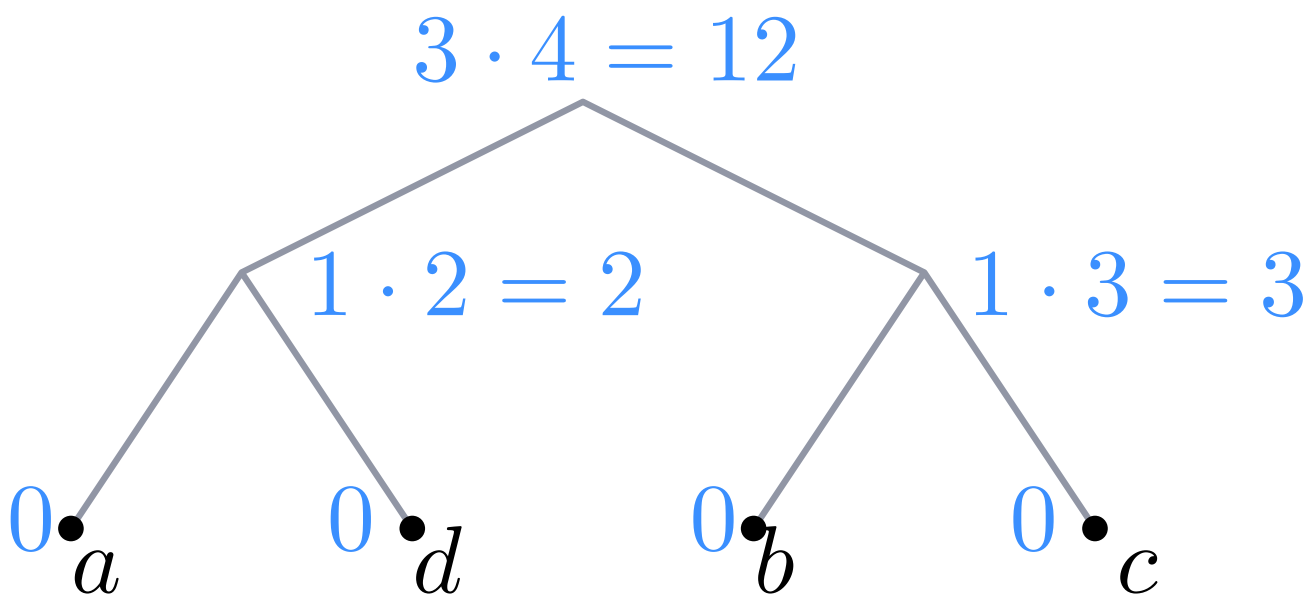
$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?



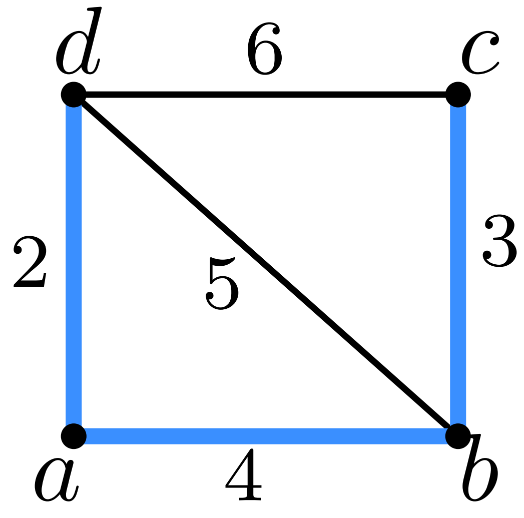
$$d_M: \begin{bmatrix} 0 & 4 & 7 & 2 \\ 4 & 0 & 3 & 5 \\ 7 & 3 & 0 & 6 \\ 2 & 5 & 6 & 0 \end{bmatrix} \qquad d_{HST}: \begin{bmatrix} 0 & 12 & 12 & 2 \\ 12 & 0 & 3 & 12 \\ 12 & 3 & 0 & 12 \\ 2 & 12 & 12 & 0 \end{bmatrix}$$



$$\Delta_v := (|P_v| - 1)\, w$$

Small Assignment

Construct the HST for this metric space M (weights shown on edges). Is it a t-approximation of M?

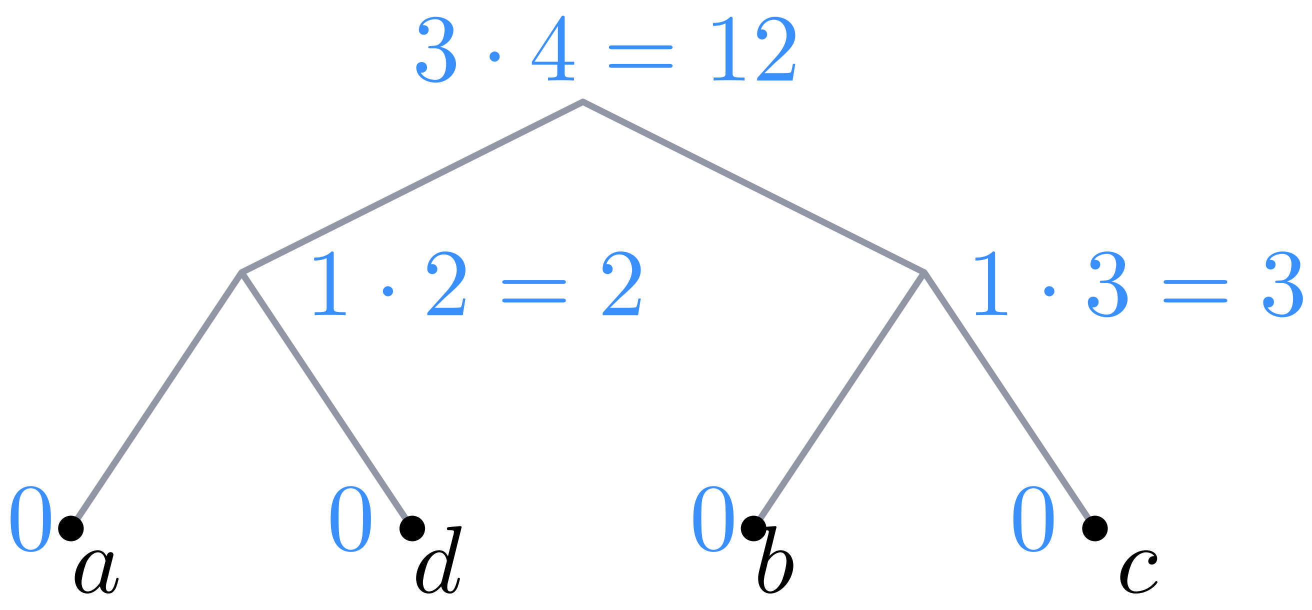


d_M :

$$\begin{bmatrix} 0 & 4 & 7 & 2 \\ 4 & 0 & 3 & 5 \\ 7 & 3 & 0 & 6 \\ 2 & 5 & 6 & 0 \end{bmatrix}$$

d_{HST} :

$$\begin{bmatrix} 0 & 12 & 12 & 2 \\ 12 & 0 & 3 & 12 \\ 12 & 3 & 0 & 12 \\ 2 & 12 & 12 & 0 \end{bmatrix}$$

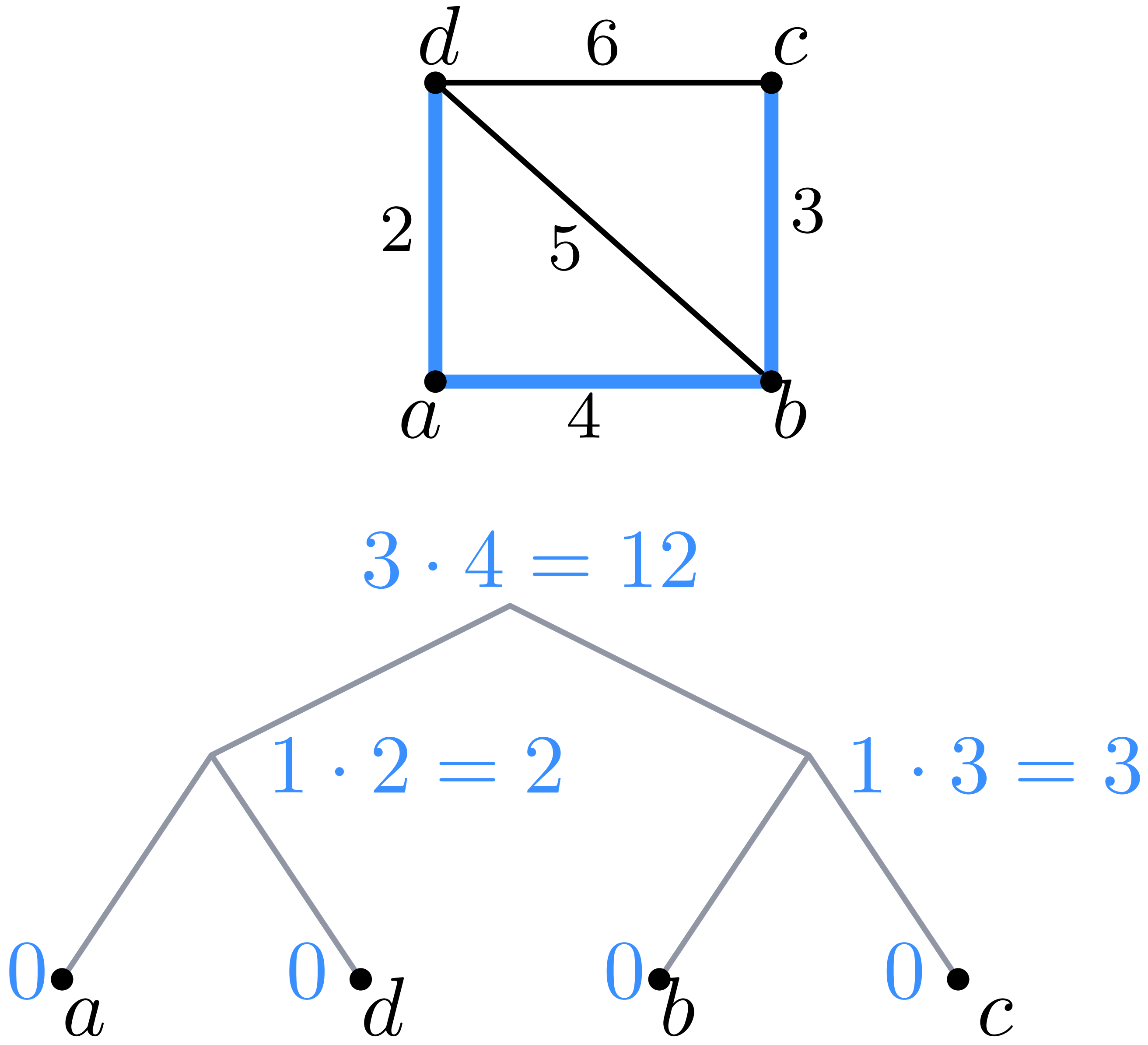


4-approximation

$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \leq d_{HST} \leq (n-1)d_M$$

(0. if u child of v: $\Delta_u \leq \Delta_v$)



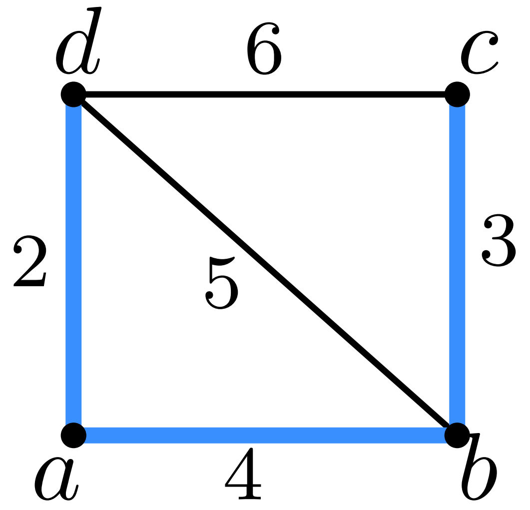
$$\Delta_v := (|P_v| - 1)\, w$$

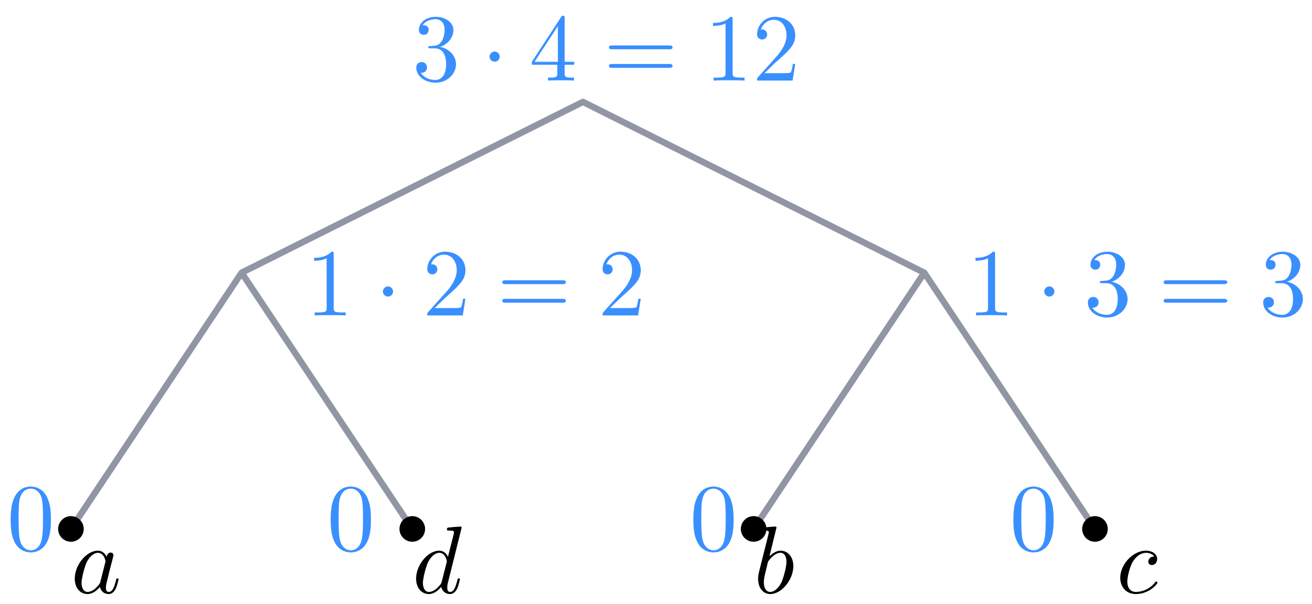
$$d_M \leq d_{HST} \leq (n-1)d_M$$

(0. if u child of v: $\Delta_u \leq \Delta_v$)

a.) $|P_u| < |P_v|$

b.) edges handled in increasing order by weight



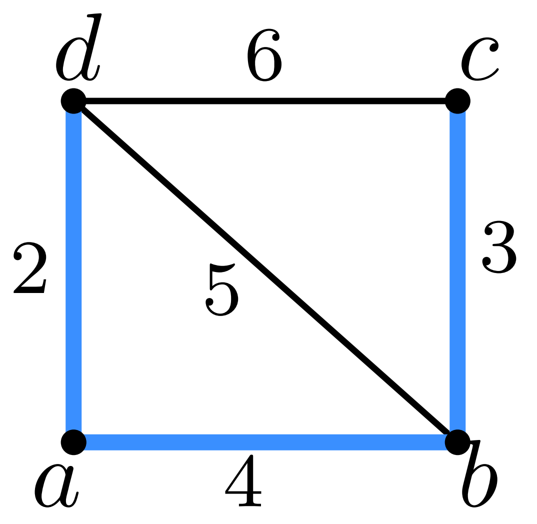


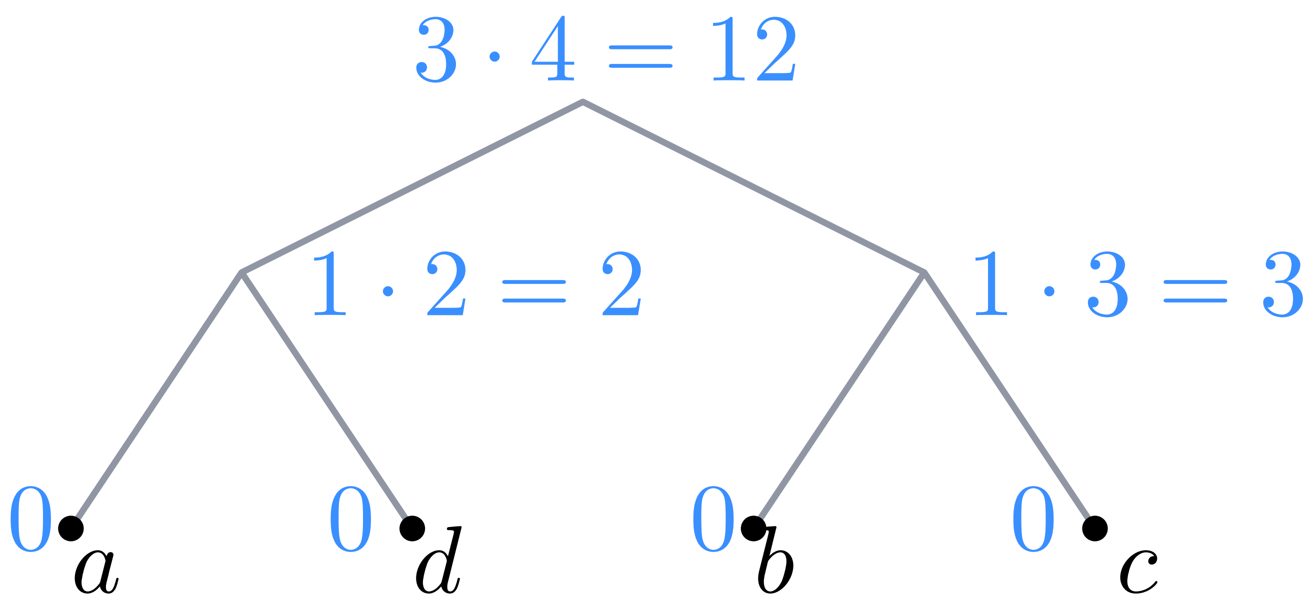
$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \leq d_{HST} \leq (n-1)d_M$$

(0. if u child of v: $\Delta_u \leq \Delta_v$)

1. $d_{HST} \leq (n-1)d_M$





$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \le d_{HST} \le (n-1)d_M$$

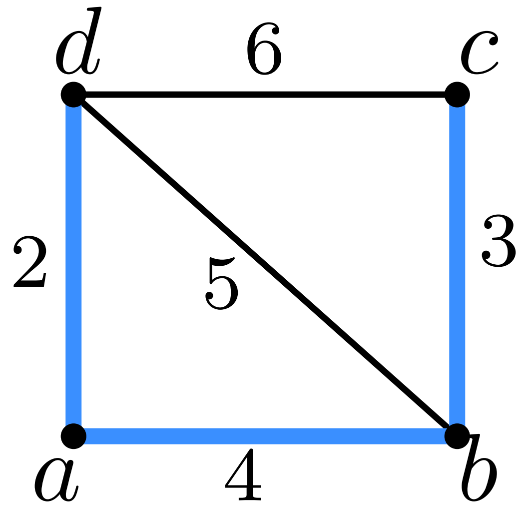
(0. if u child of v: $\Delta_u \le \Delta_v$)

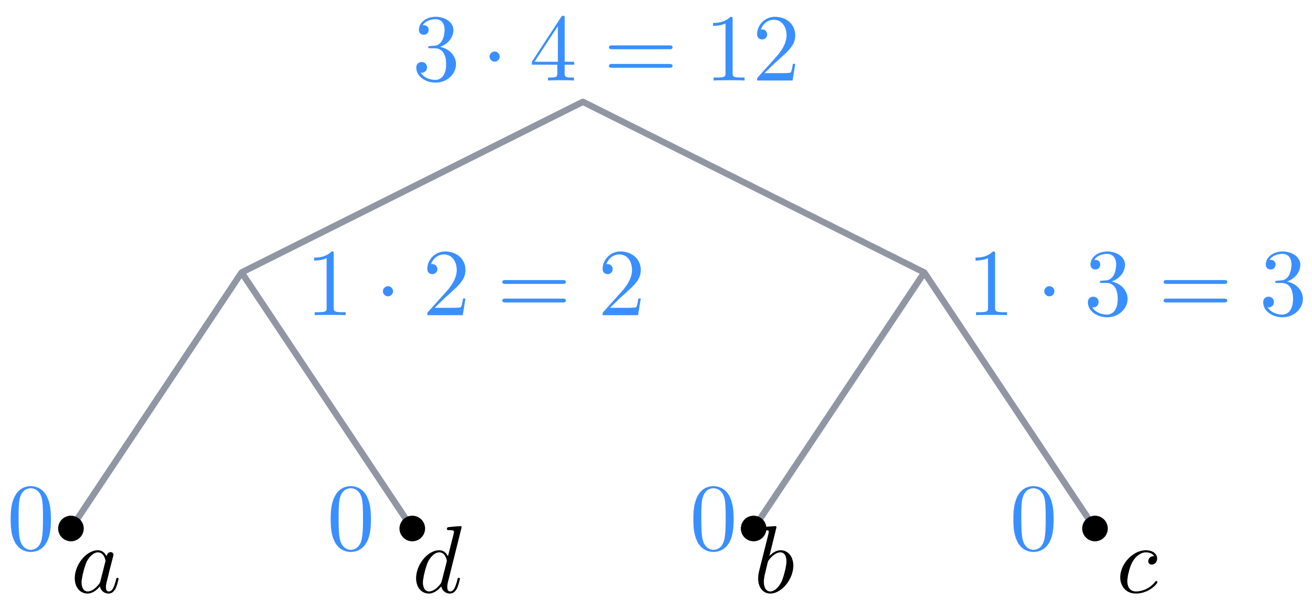
1. $d_{HST} \le (n-1)d_M$

$x, y \in P$, $v := lca(u_x, u_y)$. Then

$|P_v| - 1 \le n - 1$

weight(MST edge for v) $\le d_M(x, y)$





$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \leq d_{HST} \leq (n-1)d_M$$

(0. if u child of v: $\Delta_u \leq \Delta_v$)

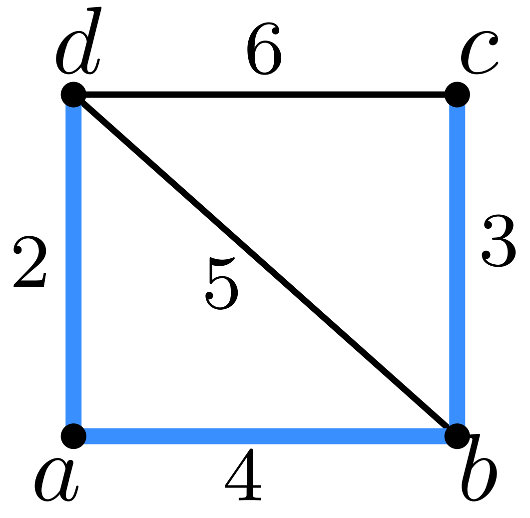
1. $d_{HST} \leq (n-1)d_M$

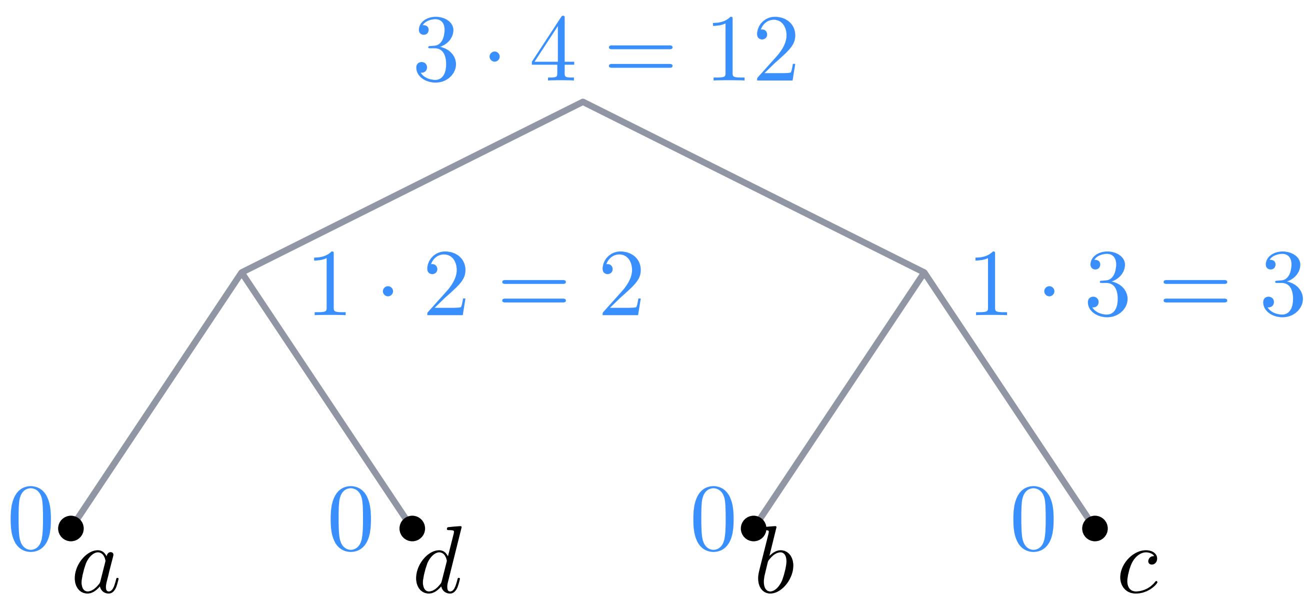
$x, y \in P$, $v := lca(u_x, u_y)$. Then

$|P_v| - 1 \leq n - 1$

weight(MST edge for v) $\leq d_M(x, y)$

$\Rightarrow d_{HST}(x, y) = \Delta_v \leq (n-1)d_M(x, y)$



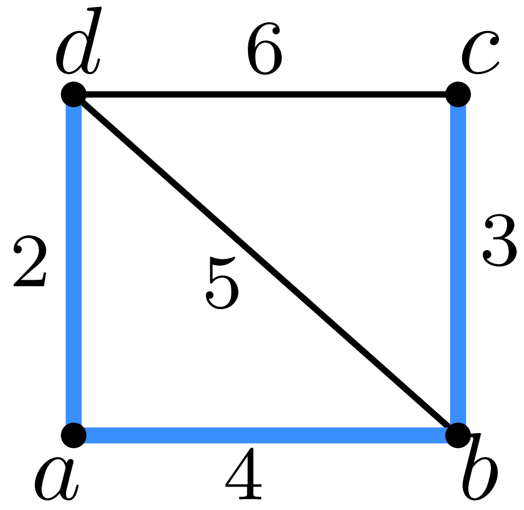


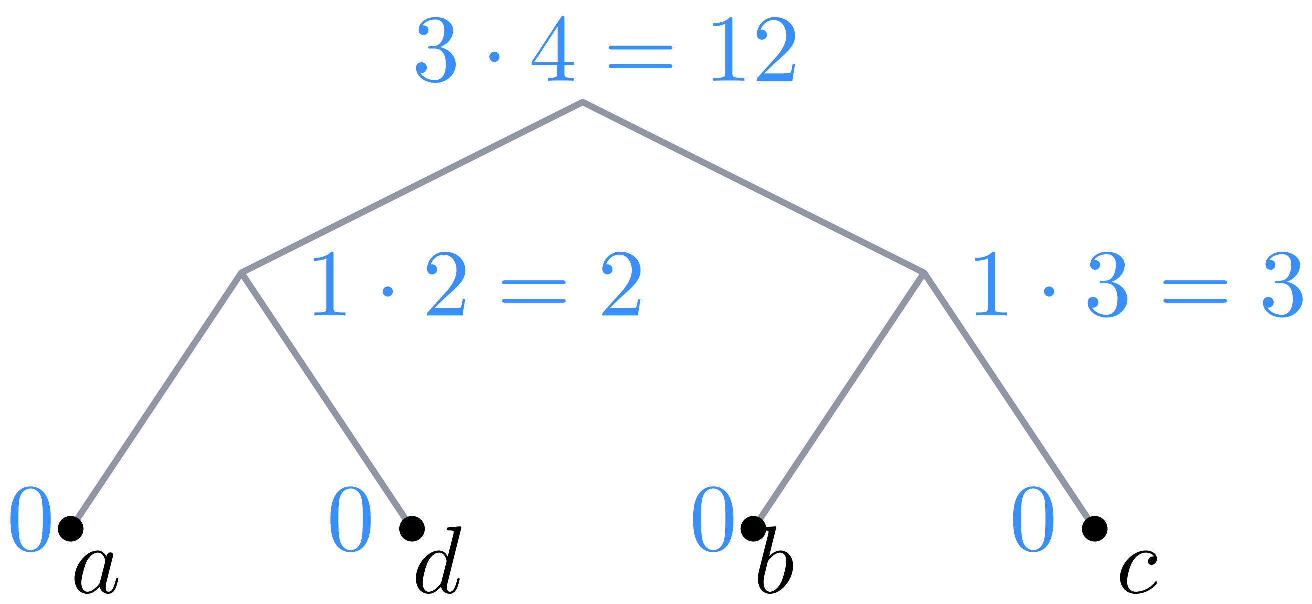
$\Delta_v := (|P_v| - 1)\, w$

$$d_M \le d_{HST} \le (n-1)d_M$$

(0. if u child of v: $\Delta_u \le \Delta_v$)

1. $d_{HST} \le (n-1)d_M$
2. $d_M \le d_{HST}$





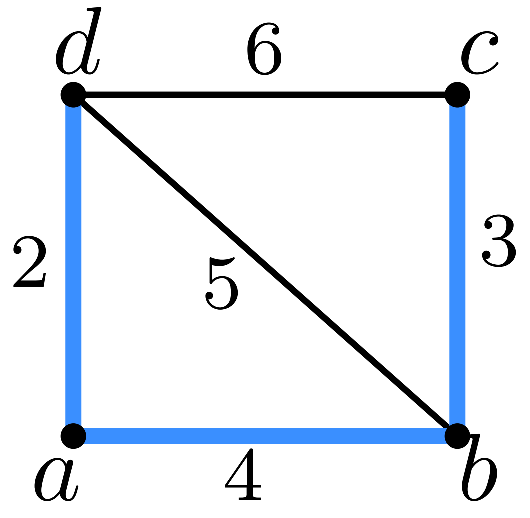
$$\Delta_v := (|P_v| - 1)\, w$$

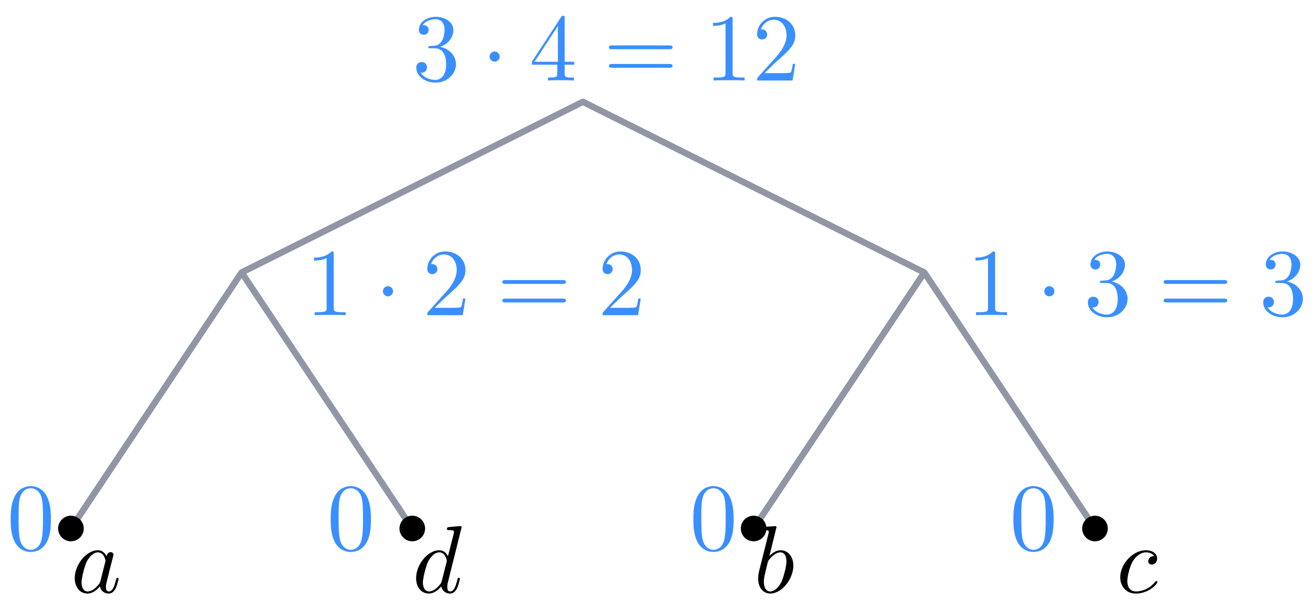
$$d_M \leq d_{HST} \leq (n-1)d_M$$

(0. if u child of v: $\Delta_u \leq \Delta_v$)

1. $d_{HST} \leq (n-1)d_M$

2. $d_M \leq d_{HST}$
   $x, y \in P$, $v := lca(u_x, u_y)$,
   $\langle e_1, \ldots, e_k \rangle$ path from x to y in MST





$$\Delta_v := (|P_v| - 1)\, w$$

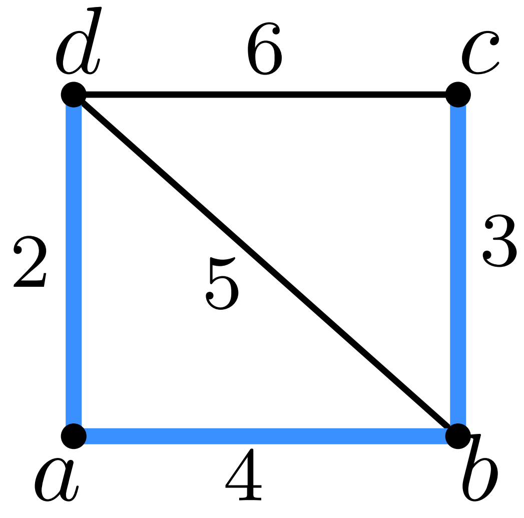
$$d_M \leq d_{HST} \leq (n-1)d_M$$

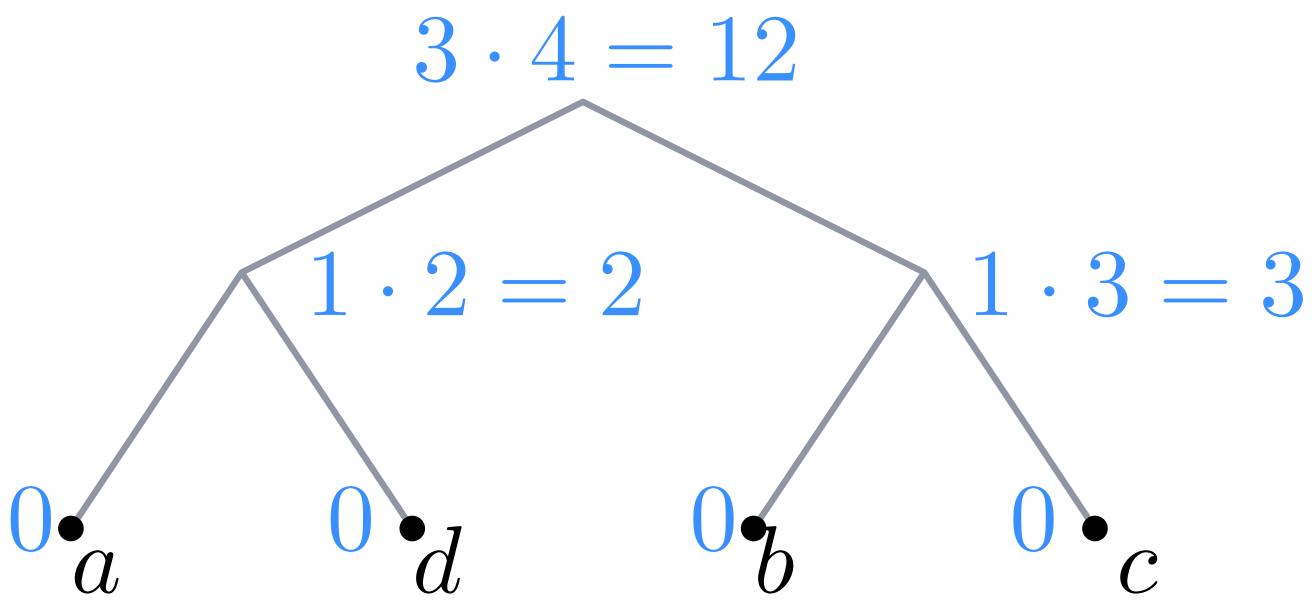
(0. if u child of v: $\Delta_u \leq \Delta_v$)

1. $d_{HST} \leq (n-1)d_M$

2. $d_M \leq d_{HST}$
   $x, y \in P$, $v := lca(u_x, u_y)$,
   $\langle e_1, \ldots, e_k \rangle$ path from x to y in MST

   $d_M(x, y)$
   = weight of shortest path from x to y in G
   $\leq \sum e_i \leq |P_v| \max e_i = \Delta_v$

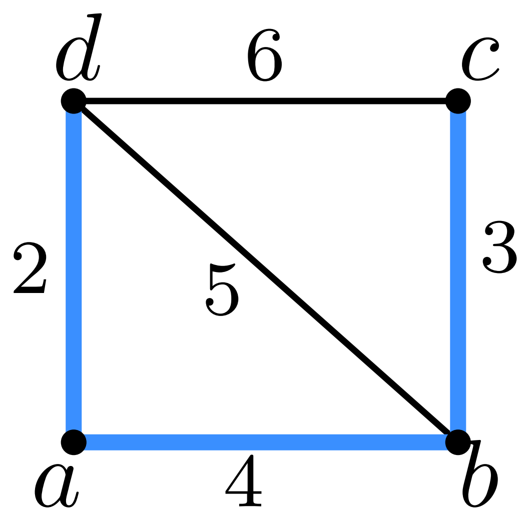


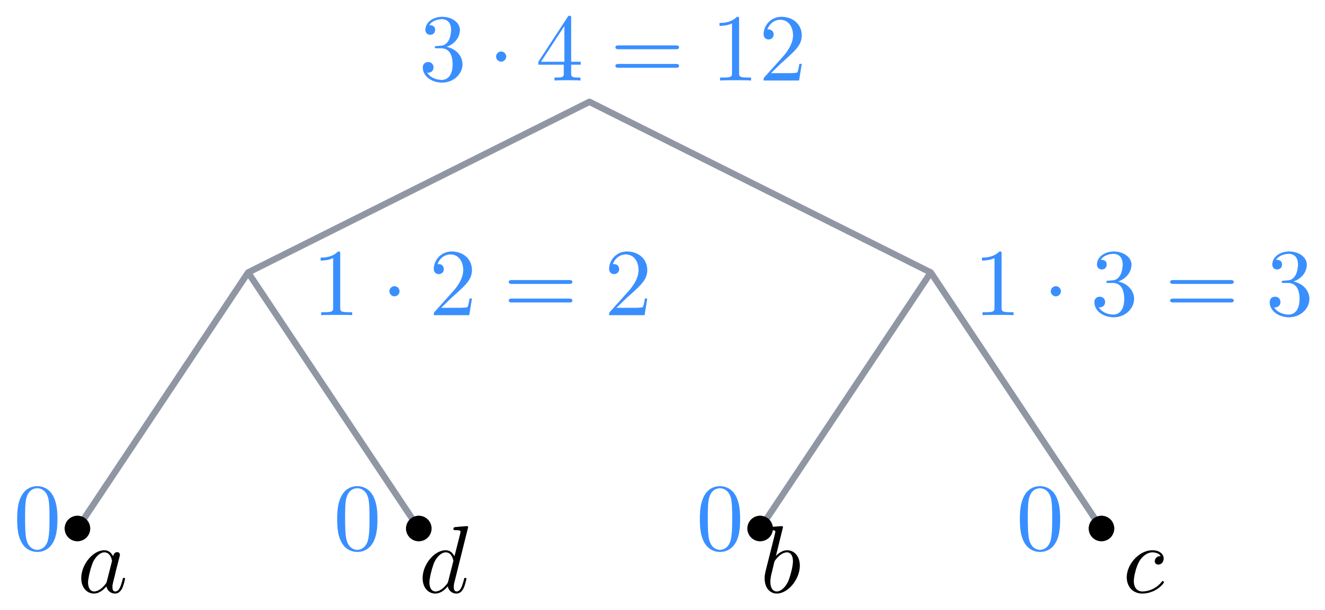


$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \leq d_{HST} \leq (n-1)d_M$$

summary:
Given a metric over a set P, we can efficiently construct a hierarchically well-balanced tree that $(n-1)$-approximates the metric.



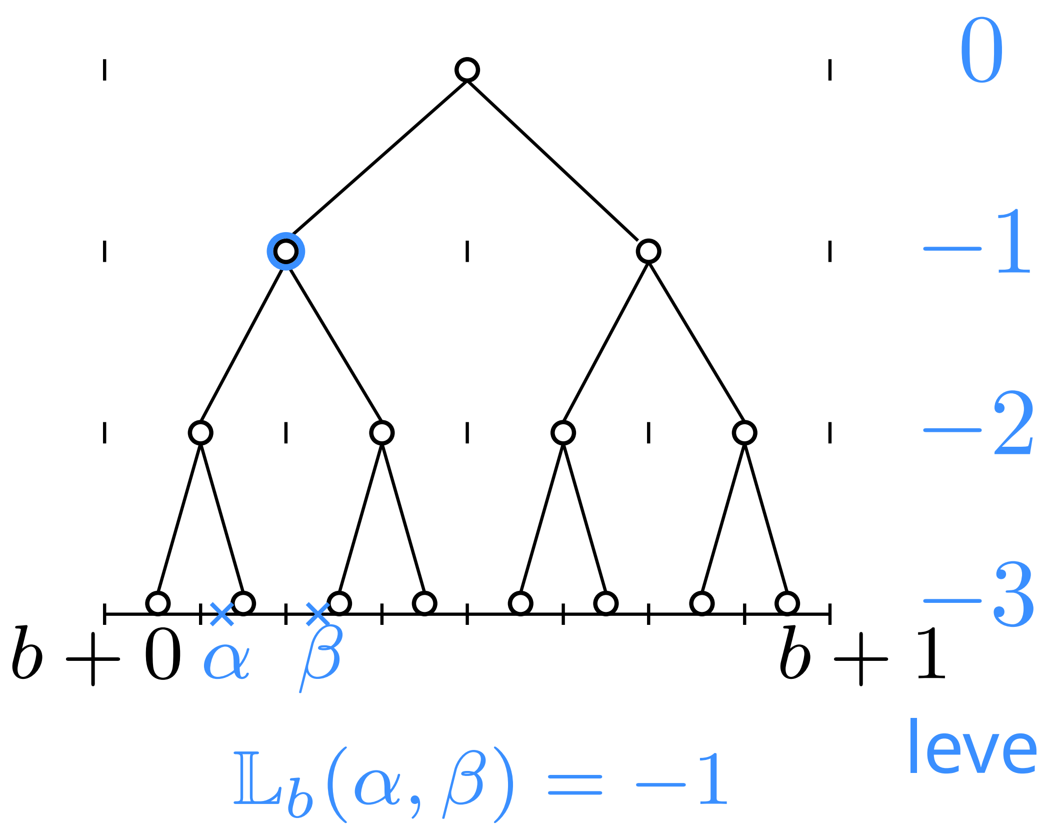


$$\Delta_v := (|P_v| - 1)\, w$$

$$d_M \leq d_{HST} \leq (n-1)d_M$$

summary:
Given a metric over a set P, we can efficiently construct a hierarchically well-balanced tree that $(n-1)$-approximates the metric.

alternative for metric space $\mathbb{R}^d$: shifted quadtree, with Δ_v = diameter of cell

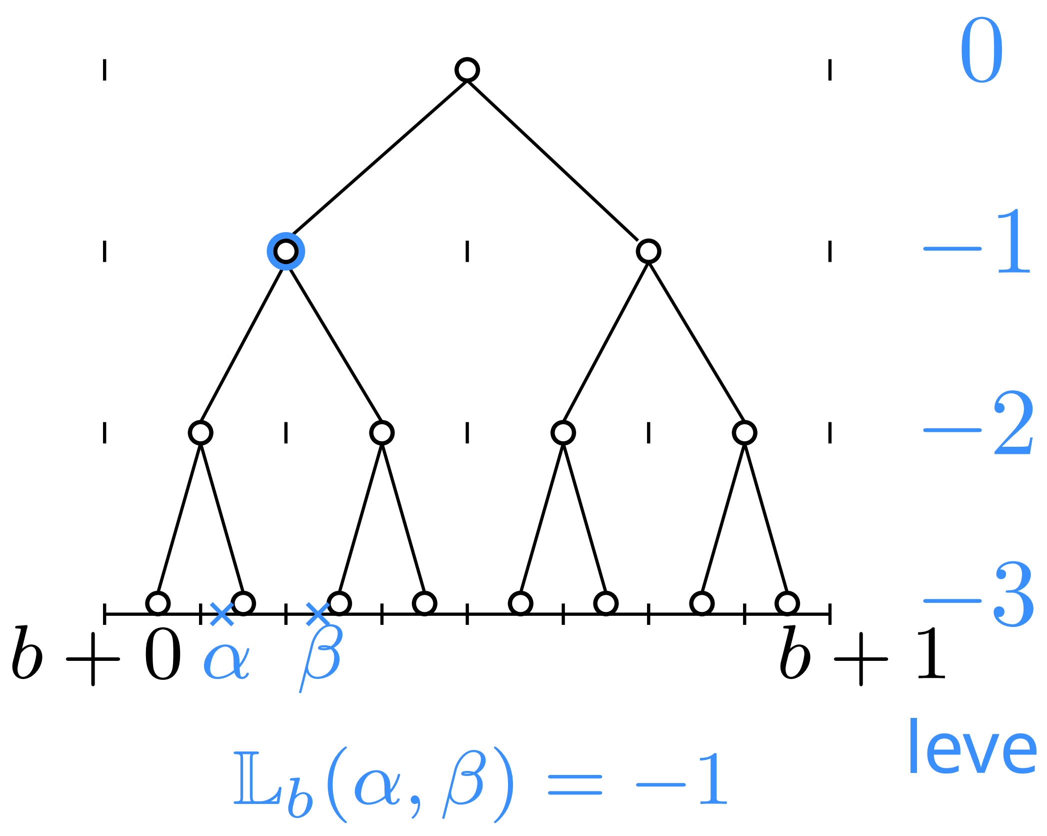


$$d_M \leq d_{HST} \leq (n-1)d_M$$

summary:
Given a metric over a set P, we can efficiently construct a hierarchically well-balanced tree that $(n-1)$-approximates the metric.

alternative for metric space $\mathbb{R}^d$: shifted quadtree, with $\Delta_v =$ diameter of cell

$$d_M(p,q) = \|p-q\| \leq \Delta_{lca(u_p,u_q)} = d_{HST}(p,q)$$



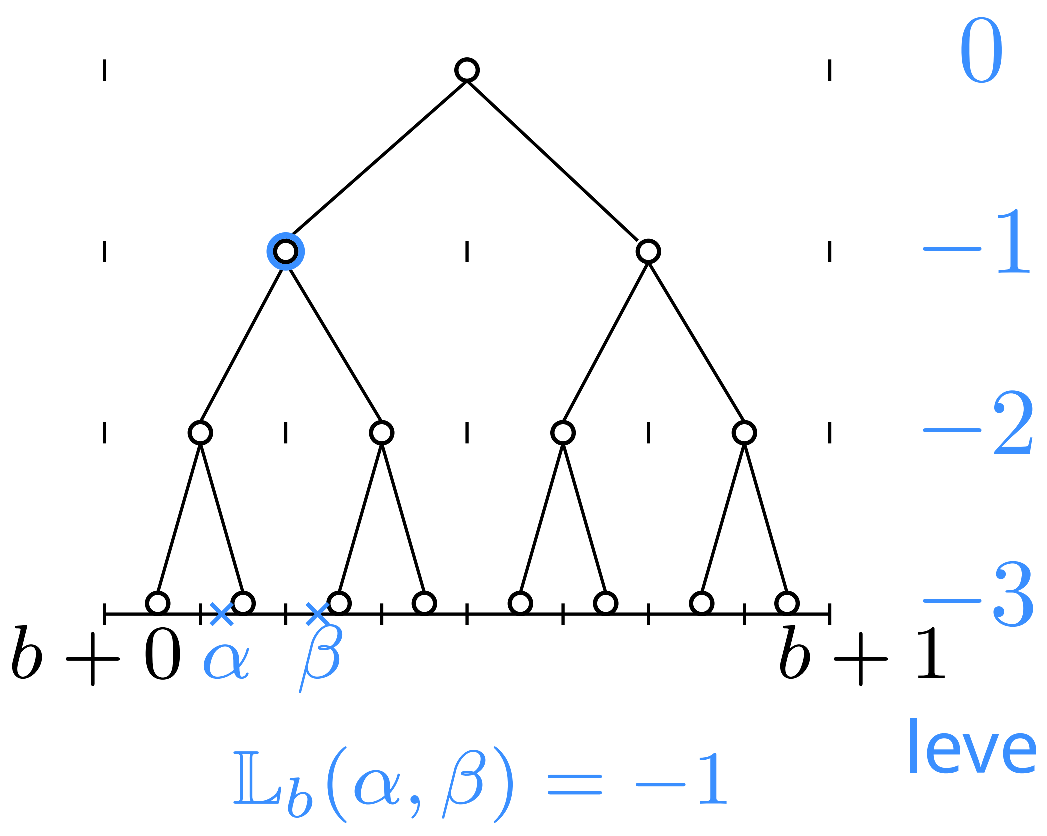
$$d_M \leq d_{HST} \leq (n-1) d_M$$

summary:
Given a metric over a set P, we can efficiently construct a hierarchically well-balanced tree that $(n-1)$-approximates the metric.

alternative for metric space $\mathbb{R}^d$: shifted quadtree, with Δ_v = diameter of cell

$$d_M(p,q) = \|p-q\| \leq \Delta_{lca(u_p,u_q)} = d_{HST}(p,q)$$

$$\leq n^{O(1)} \|p-q\| \quad \text{(with high probability)}$$



$$d_M \leq d_{HST} \leq (n-1) d_M$$

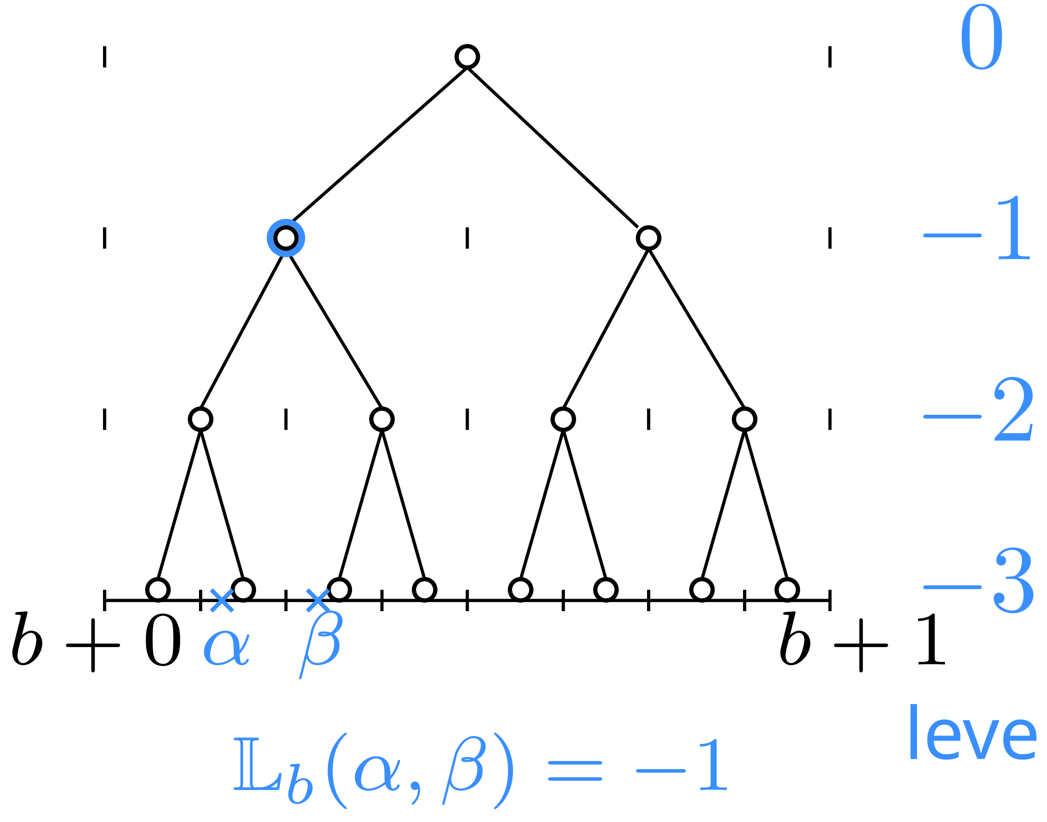
summary:
Given a metric over a set P, we can efficiently construct a hierarchically well-balanced tree that $(n-1)$-approximates the metric.

alternative for metric space $\mathbb{R}^d$: shifted quadtree, with Δ_v = diameter of cell

$$d_M(p,q) = \|p-q\| \leq \Delta_{lca(u_p,u_q)} = d_{HST}(p,q)$$

$$\leq n^{O(1)} \|p-q\| \quad \text{(with high probability)}$$

$$= n^{O(1)} d_M(p,q)$$



Overview

Approximating a metric space by a hierarchical well-separated tree (HST)

hierachical well-separated trees

simple $(n-1)$-approximation

fast $n^{O(1)}$-approximation in $\mathbb{R}^d$

ANN via point location among balls

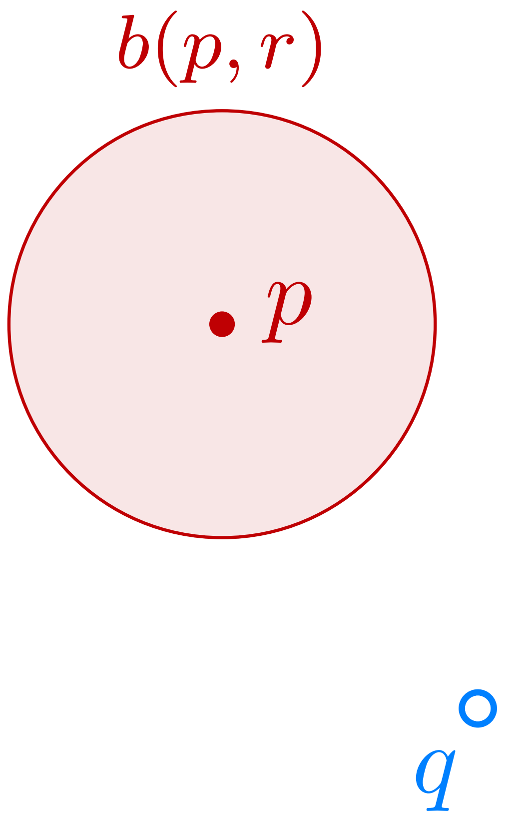
simple construction

handling a range of radii

ANN data structure based on HST

Point Location among balls

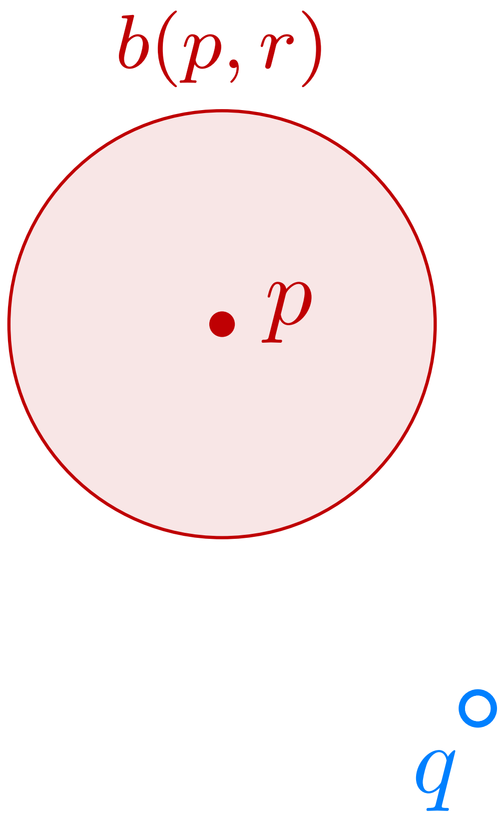
$b(p, r) = \{q \mid d(p, q) \leq r\}$ denotes a ball around point p with radius r



Point Location among balls

$b(p,r) = \{q \mid d(p,q) \leq r\}$ denotes a ball around point p with radius r

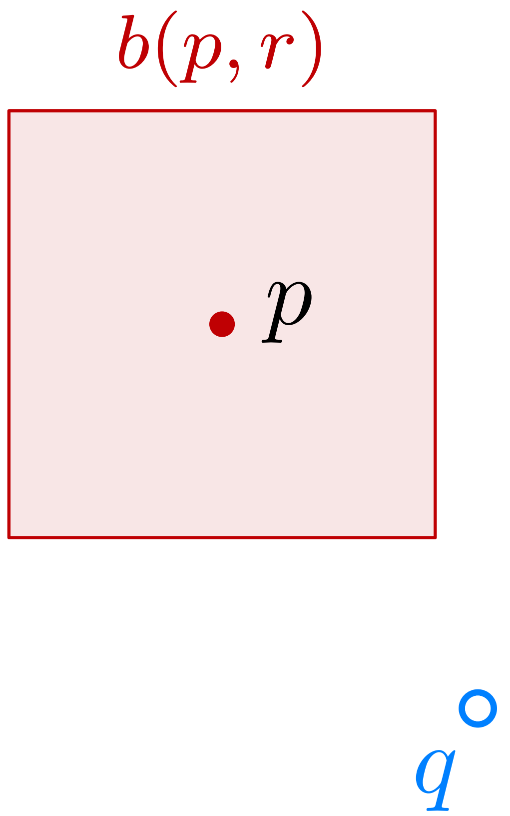
point set P in a *metric space* $\mathcal{M}$



Point Location among balls

$b(p, r) = \{q \mid d(p, q) \leq r\}$ denotes a ball around point p with radius r

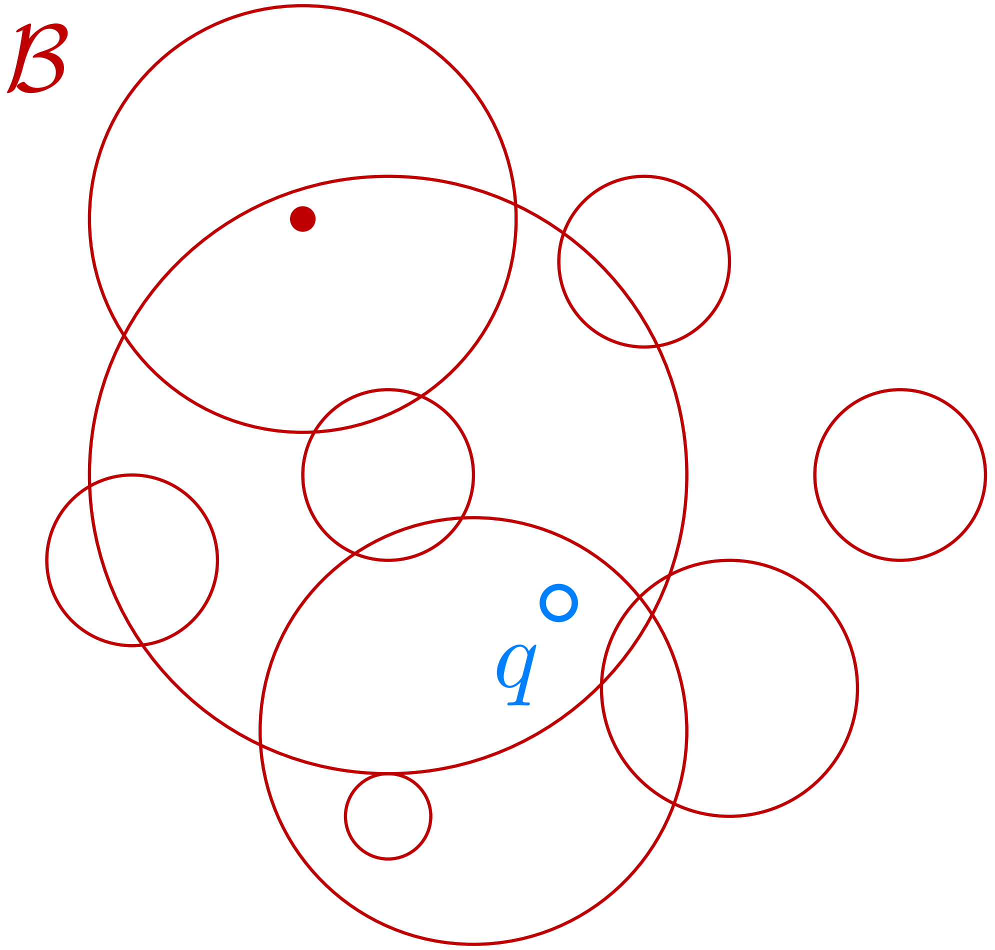
point set P in a *metric space* $\mathcal{M}$



Point Location among balls

$b(p, r) = \{q \mid d(p, q) \leq r\}$ denotes a ball around point p with radius r

For a collection of balls $\mathcal{B}$, $\odot_{\mathcal{B}}(q)$ denotes the smallest ball containing q, or target ball for q

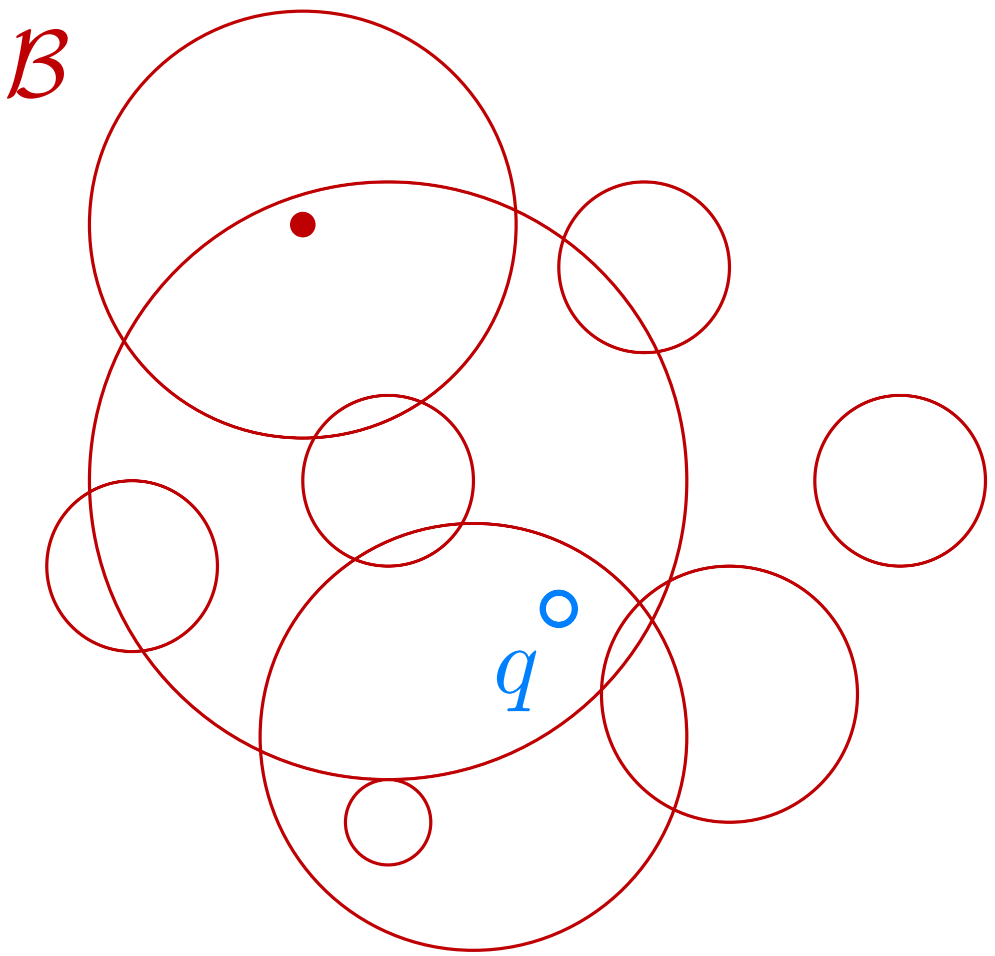


Point Location among balls

$b(p, r) = \{q \mid d(p, q) \leq r\}$ denotes a ball around point p with radius r

For a collection of balls $\mathcal{B}$, $\odot_{\mathcal{B}}(q)$ denotes the smallest ball containing q, or target ball for q

Point location among balls:
Given a collection of balls $\mathcal{B}$ and a query point q, find the target ball $\odot_{\mathcal{B}}(q)$ of q

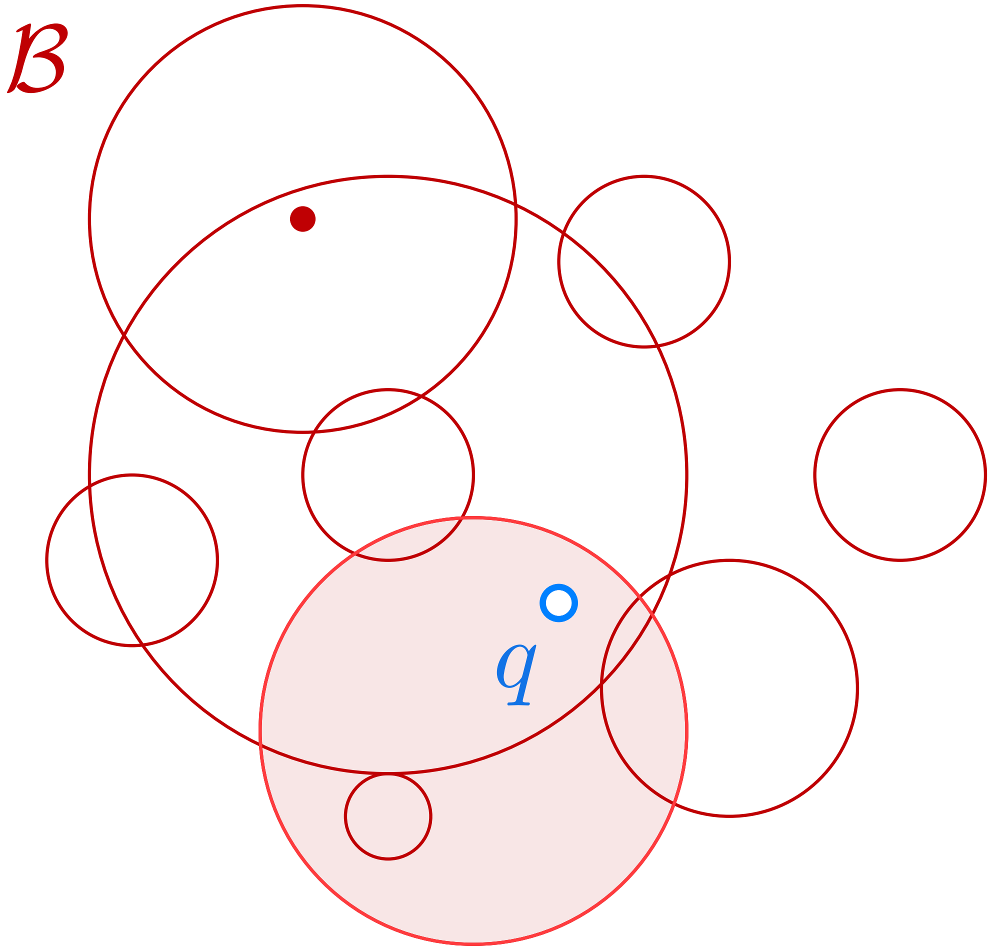


Point Location among balls

$b(p, r) = \{q \mid d(p, q) \leq r\}$ denotes a ball around point p with radius r

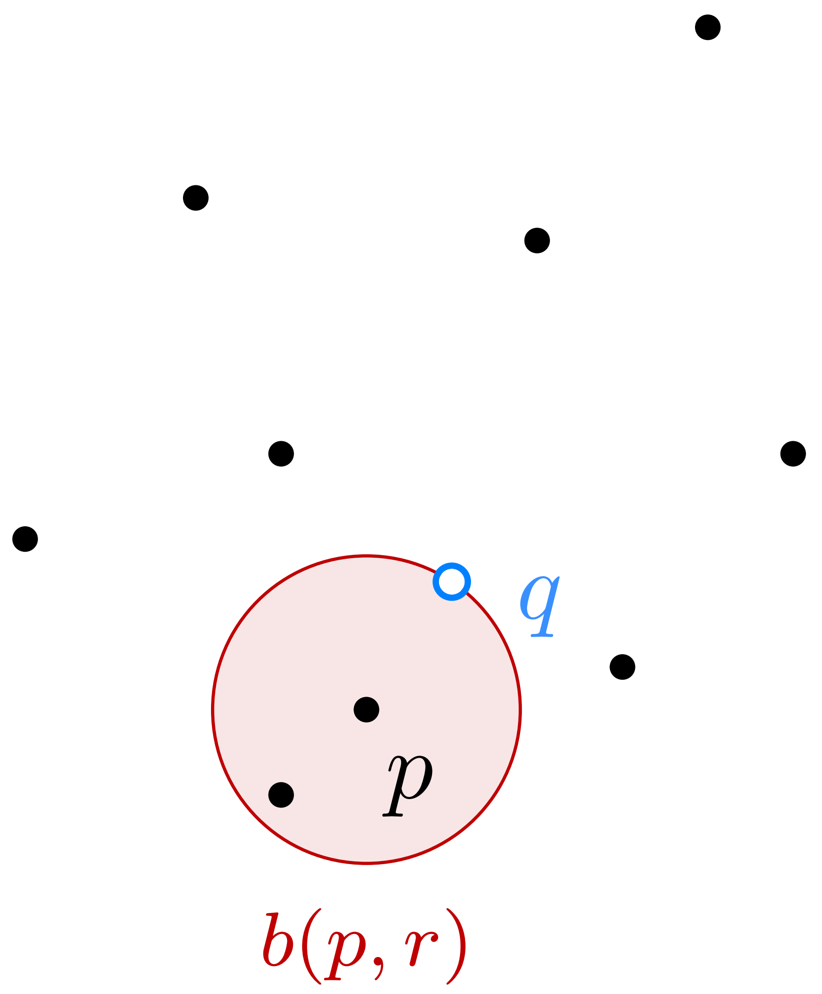
For a collection of balls $\mathcal{B}$, $\odot_{\mathcal{B}}(q)$ denotes the smallest ball containing q, or target ball for q

Point location among balls:
Given a collection of balls $\mathcal{B}$ and a query point q, find the target ball $\odot_{\mathcal{B}}(q)$ of q



Relation $(1+\varepsilon)$-**ANN** and **PLEB**

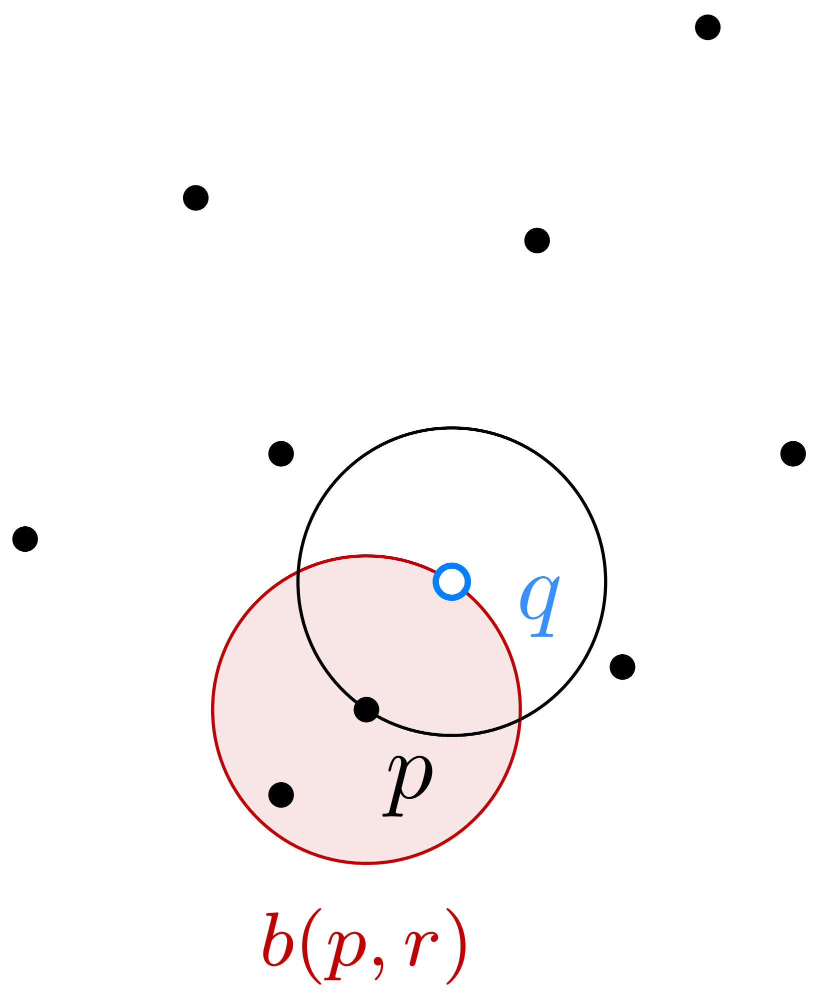
Gaining some intuition:



Relation $(1+\varepsilon)$-ANN and PLEB

Gaining some intuition:

$$r = d(q,p) \geq d(q,P)$$

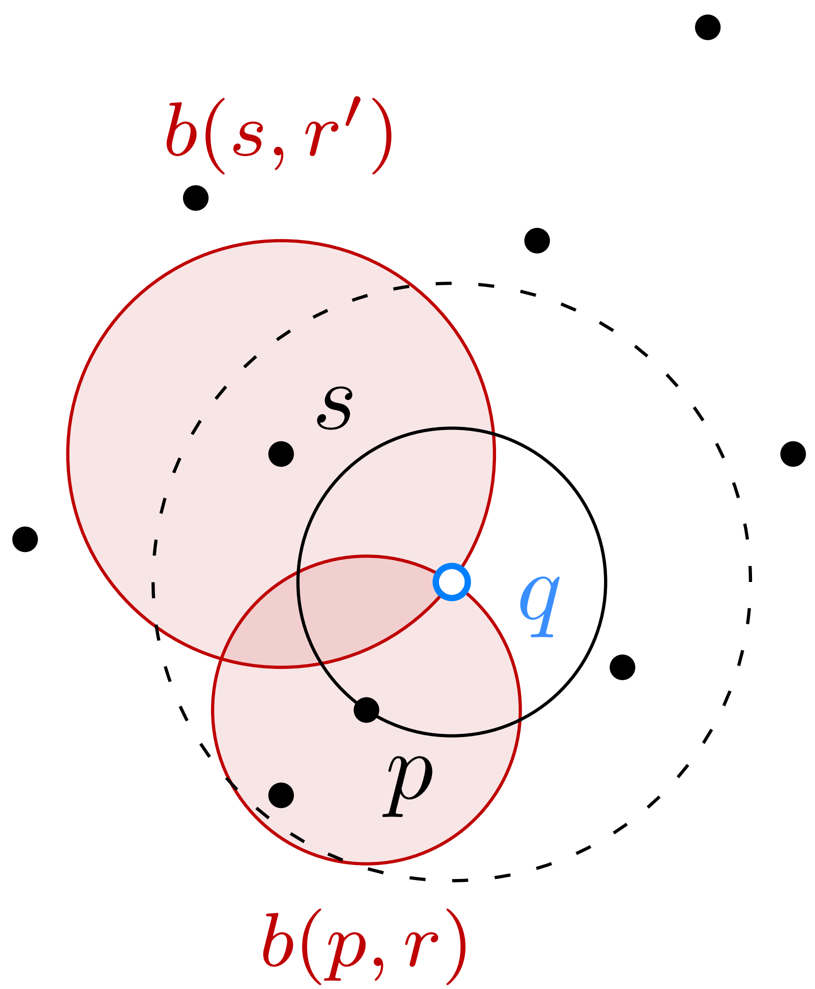


Relation $(1+\varepsilon)$-**ANN** and **PLEB**

Gaining some intuition:

$$r = d(q,p) \geq d(q,P)$$

$$r' = d(q,s) \leq (1+\varepsilon)d(q,P)$$



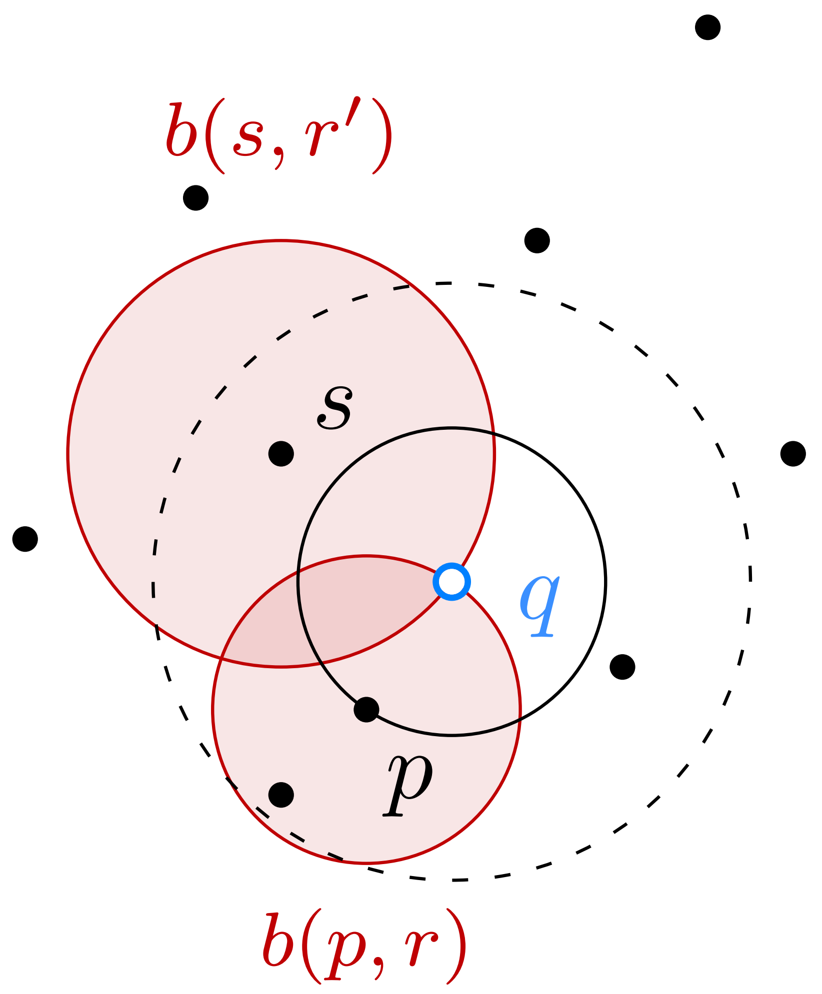
Relation $(1+\varepsilon)$-**ANN** and **PLEB**

Gaining some intuition:

$r = d(q,p) \geq d(q,P)$

$r' = d(q,s) \leq (1+\varepsilon)d(q,P)$

since $r \leq r'$, we have $d(q,p) \leq (1+\varepsilon)d(q,P)$



Relation $(1+\varepsilon)$-**ANN** and **PLEB**

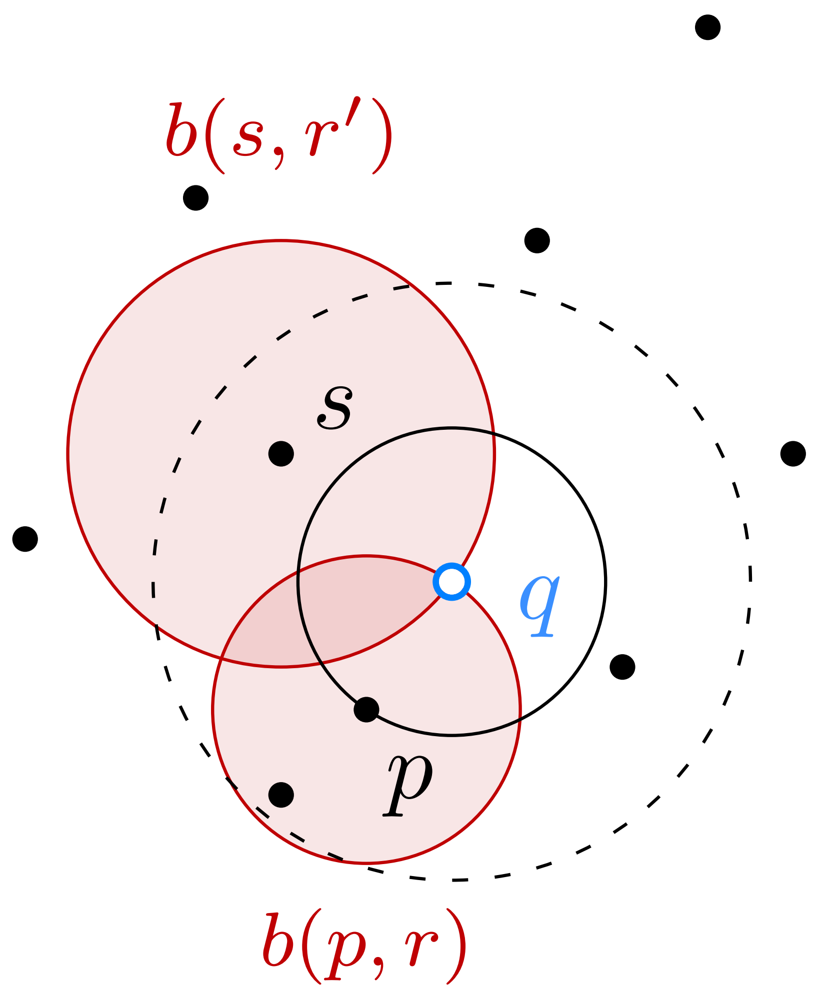
Gaining some intuition:

$r = d(q,p) \geq d(q,P)$

$r' = d(q,s) \leq (1+\varepsilon)d(q,P)$

since $r \leq r'$, we have $d(q,p) \leq (1+\varepsilon)d(q,P)$

Choice of ball sizes matters!



Relation $(1+\varepsilon)$-**ANN** and **PLEB**

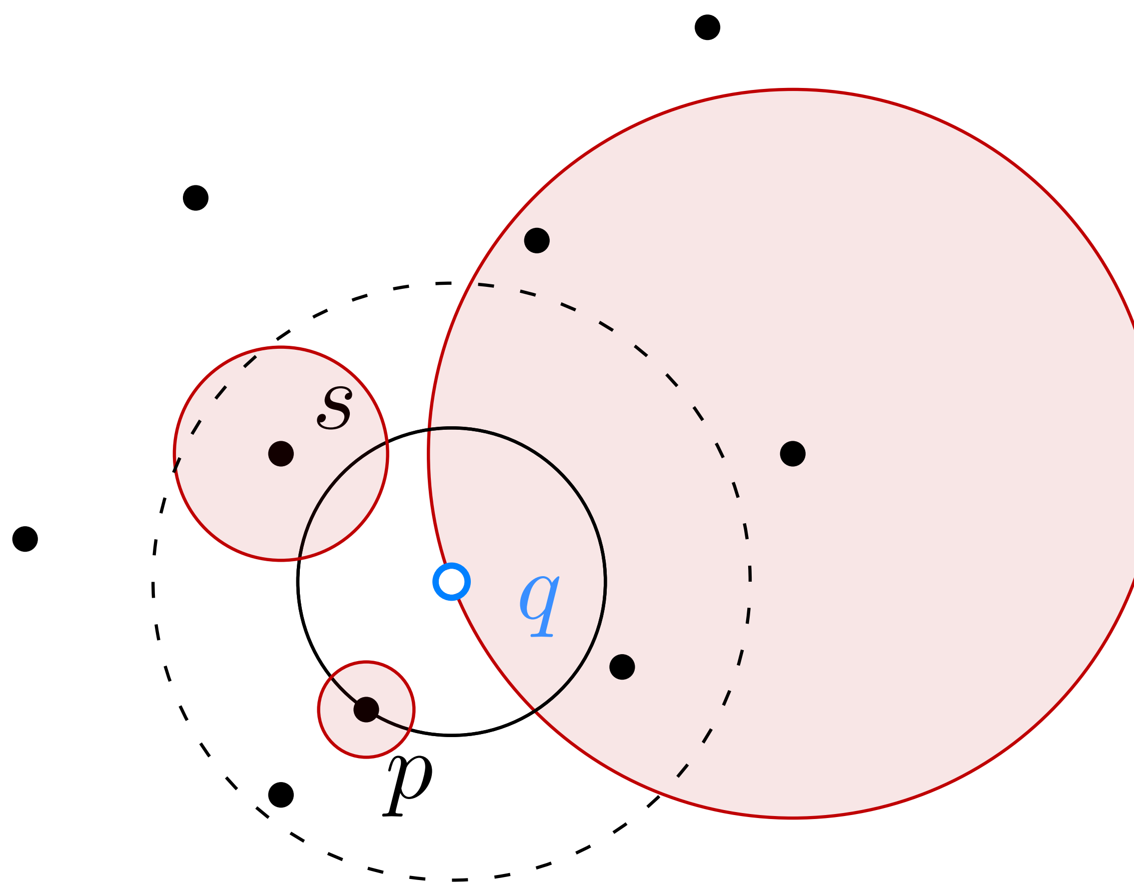
Gaining some intuition:

$r = d(q,p) \geq d(q,P)$

$r' = d(q,s) \leq (1+\varepsilon)d(q,P)$

since $r \leq r'$, we have $d(q,p) \leq (1+\varepsilon)d(q,P)$

Choice of ball sizes matters!

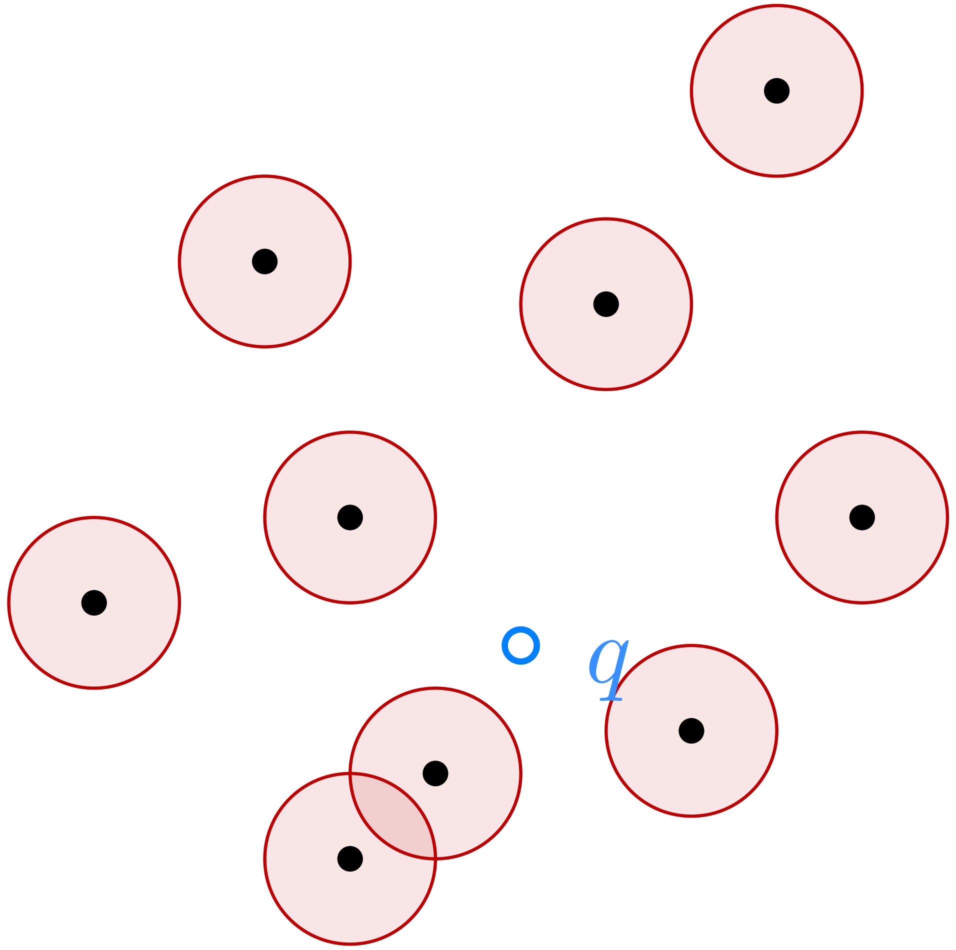


Reduction

Reduction from $(1+\varepsilon)$**-ANN** to **Point location among balls**

define $\mathcal{U}(P,r) = \bigcup_{p \in P} b(p,r)$

union of balls of radius r

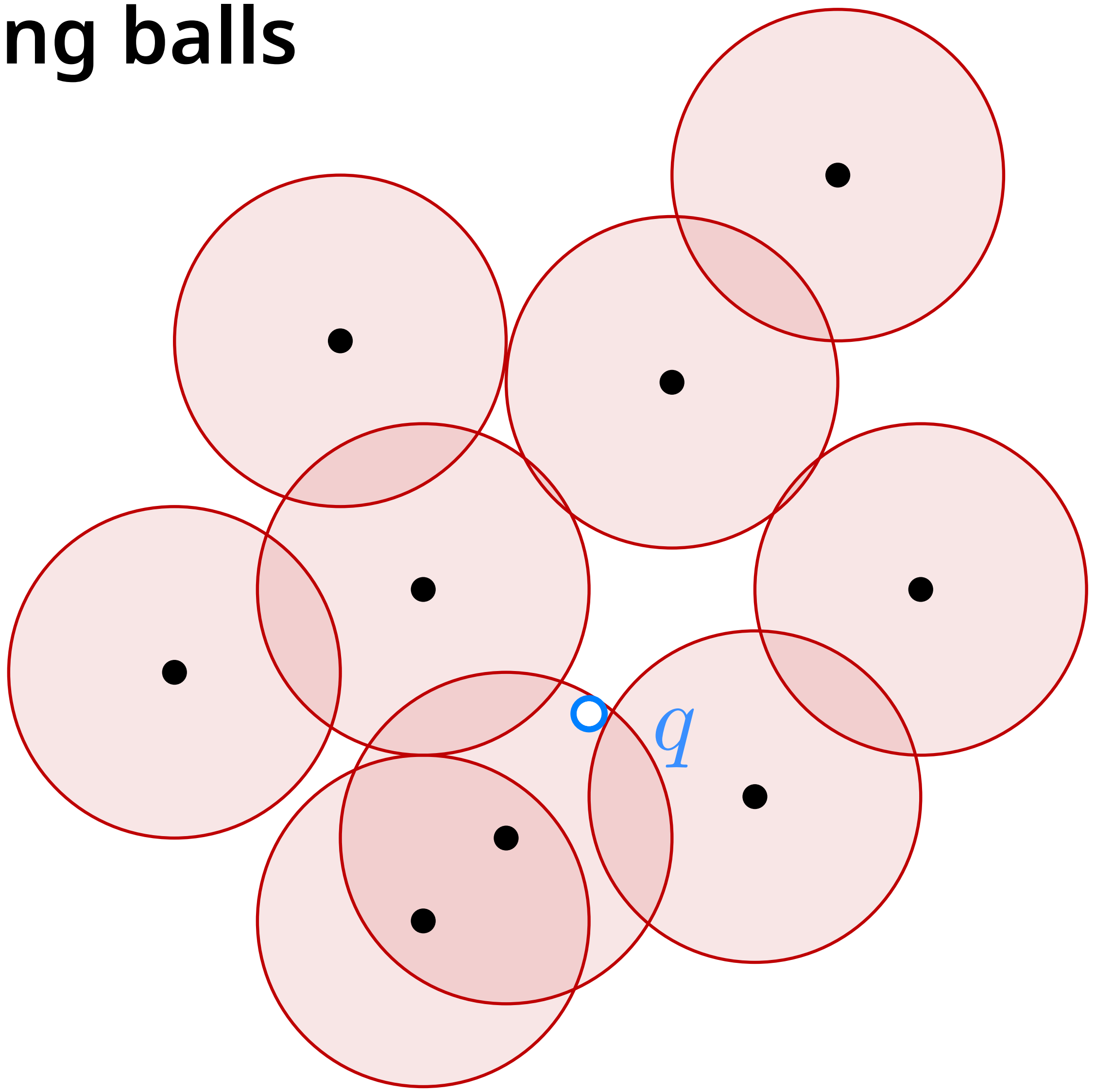


Reduction

Reduction from $(1+\varepsilon)$-**ANN** to **Point location among balls**

define $\mathcal{U}(P,r) = \bigcup_{p\in P} b(p,r)$

union of balls of radius r



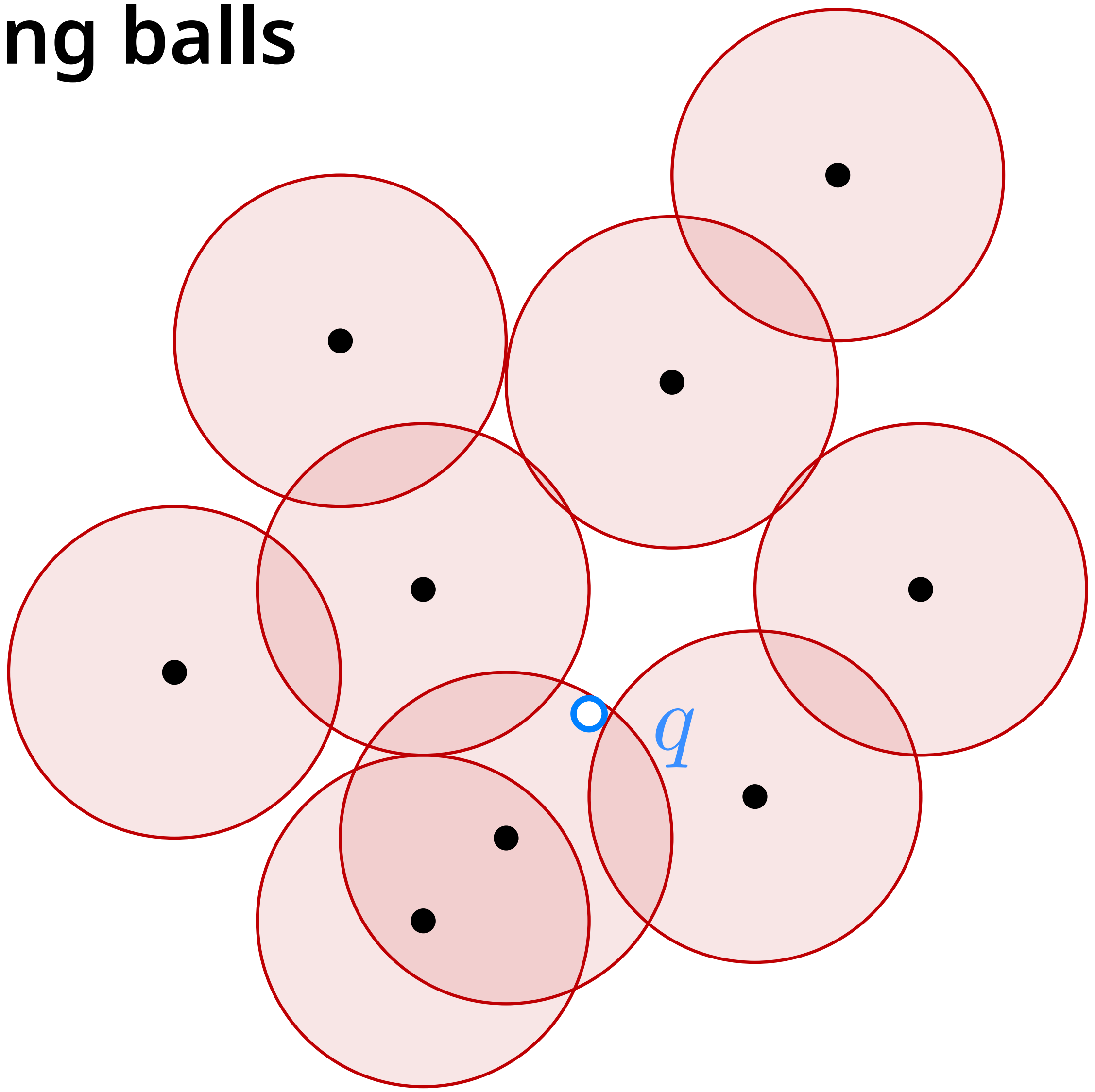
Reduction

Reduction from $(1+\varepsilon)$-**ANN** to **Point location among balls**

define $\mathcal{U}(P,r) = \bigcup_{p\in P} b(p,r)$

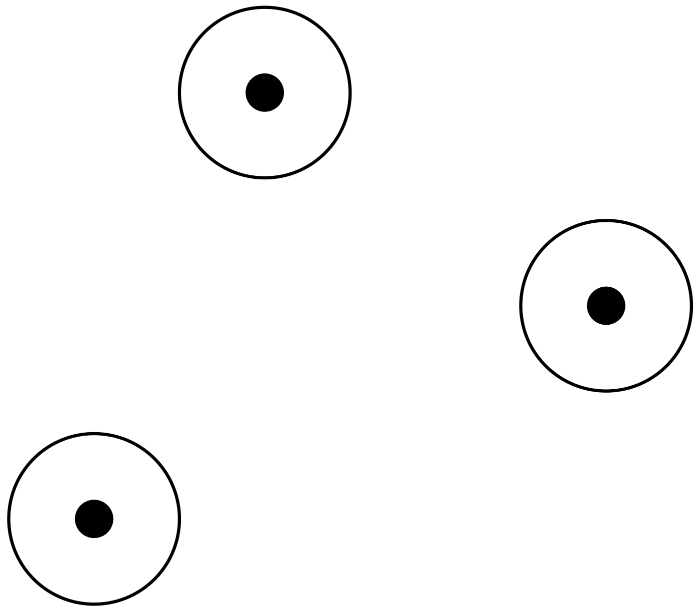
union of balls of radius r

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P,(1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



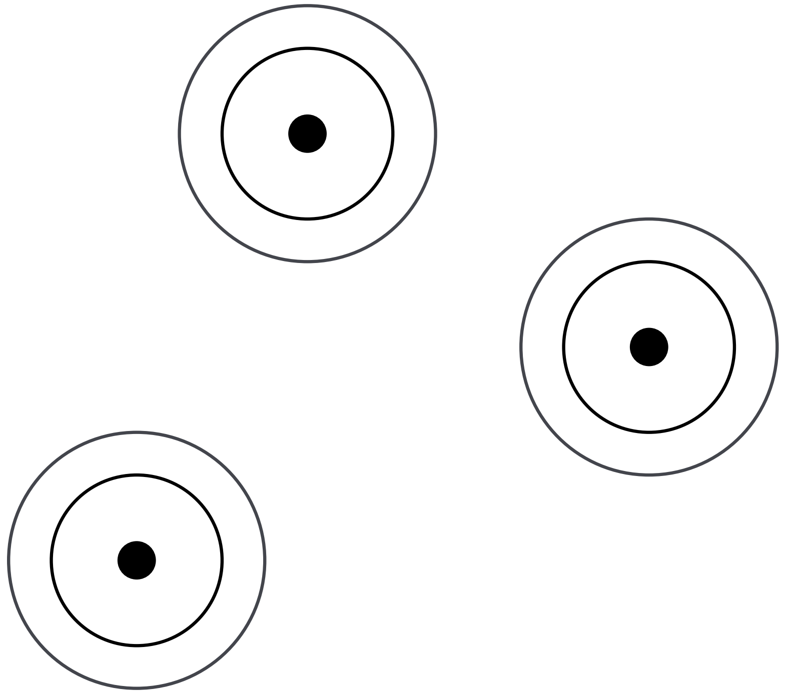
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



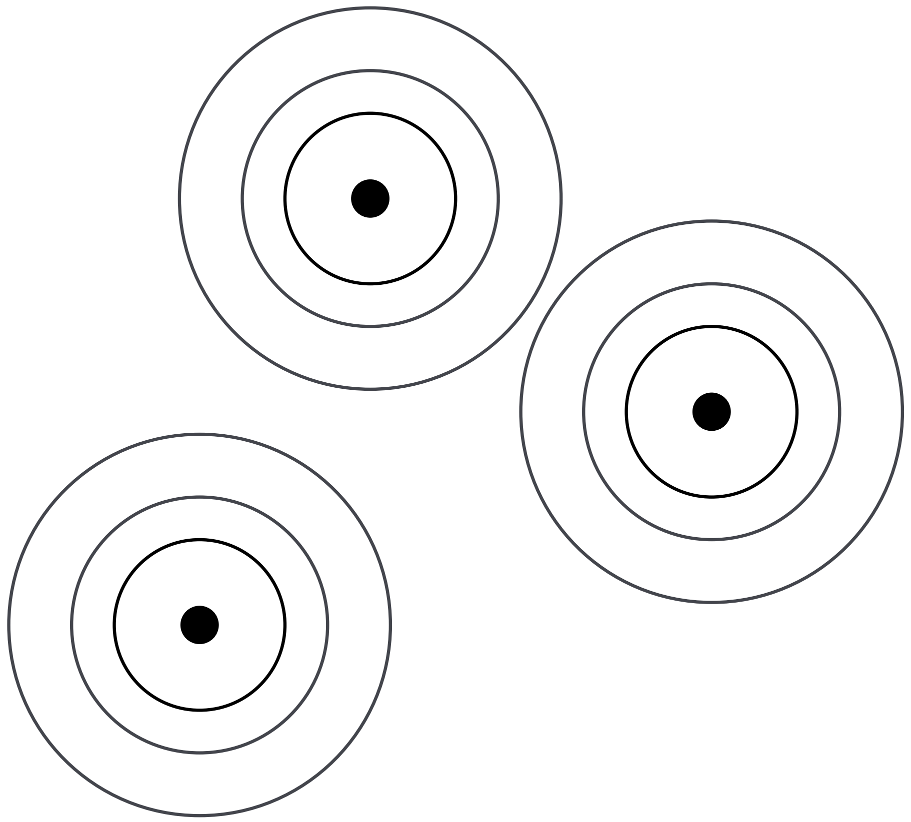
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



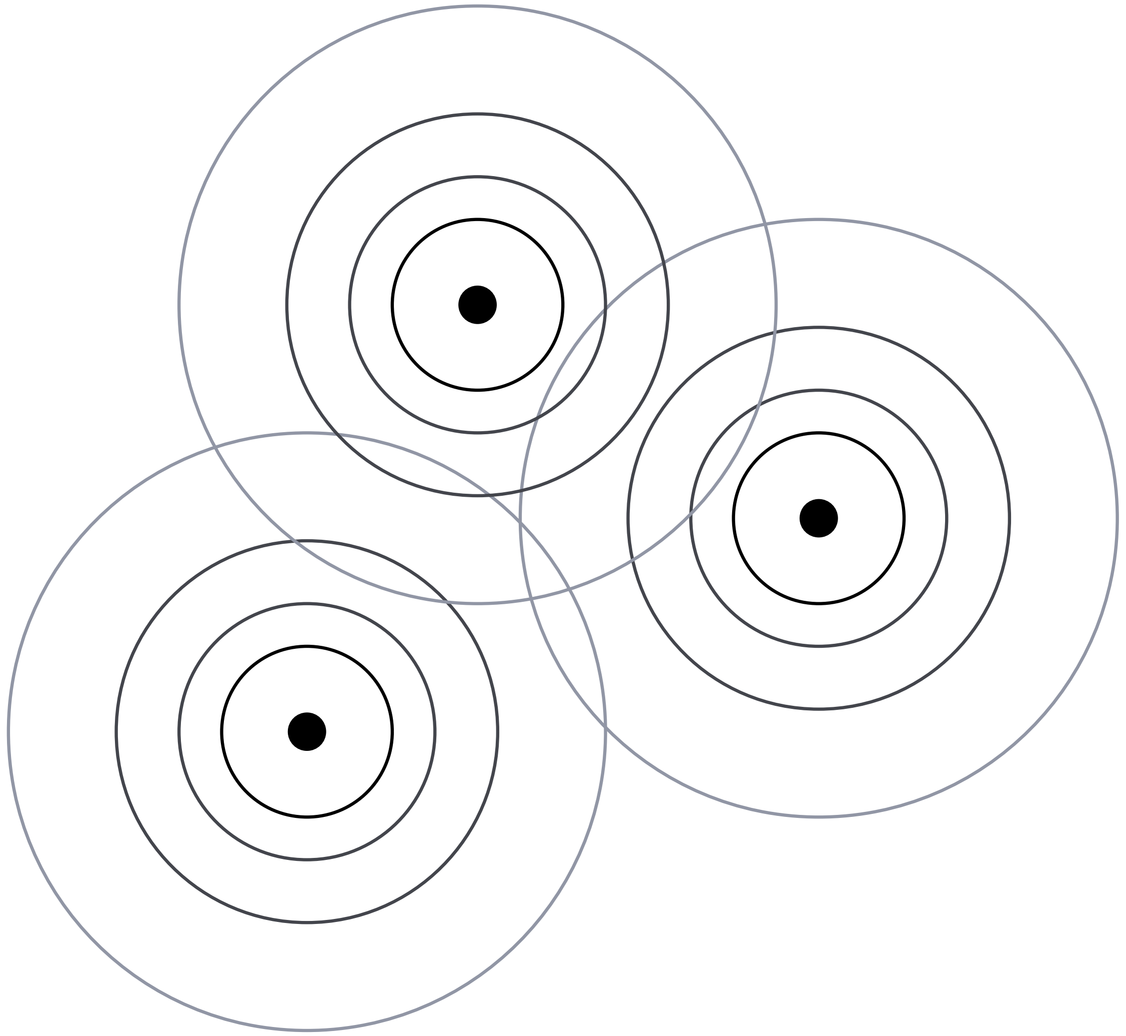
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



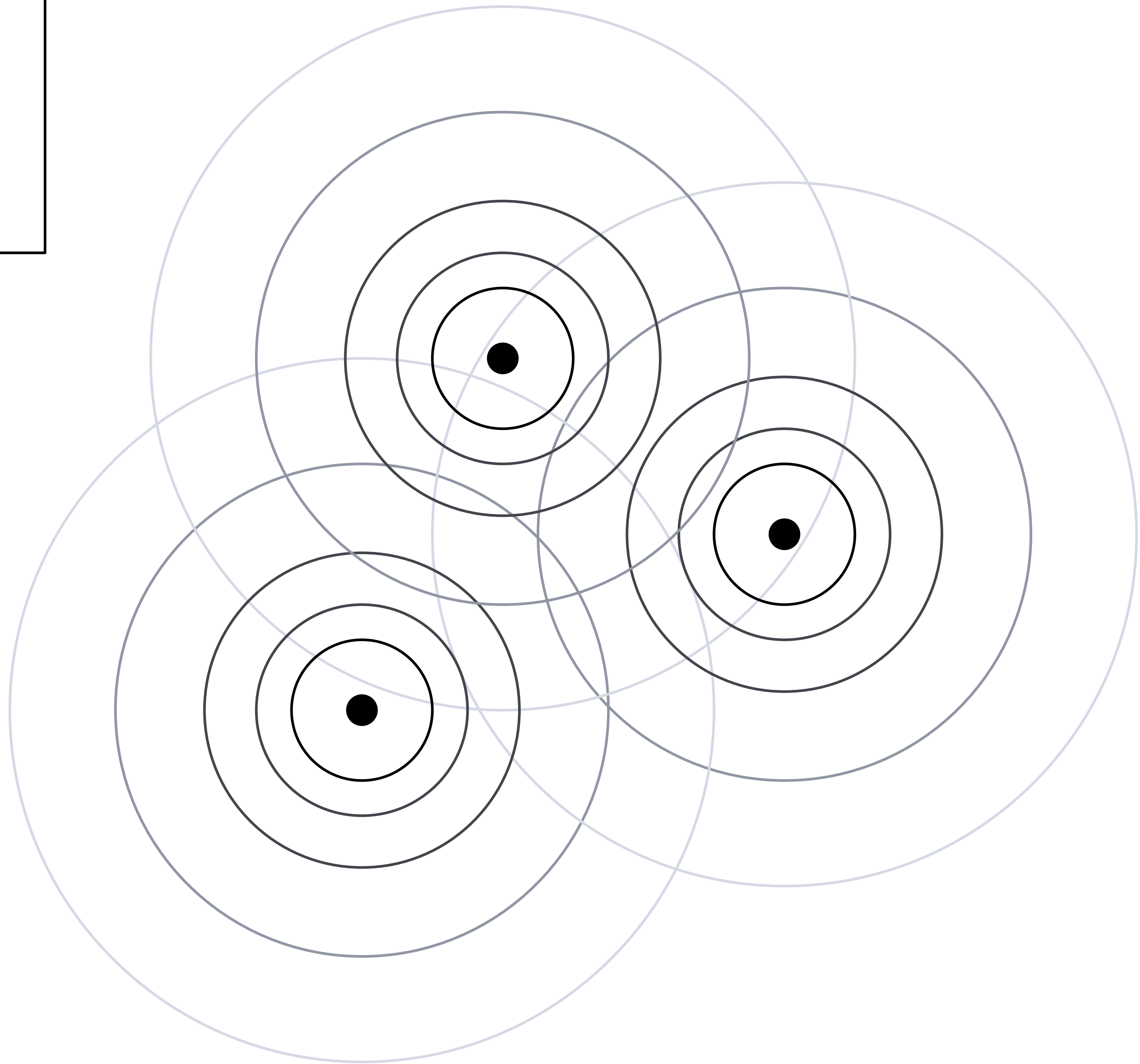
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



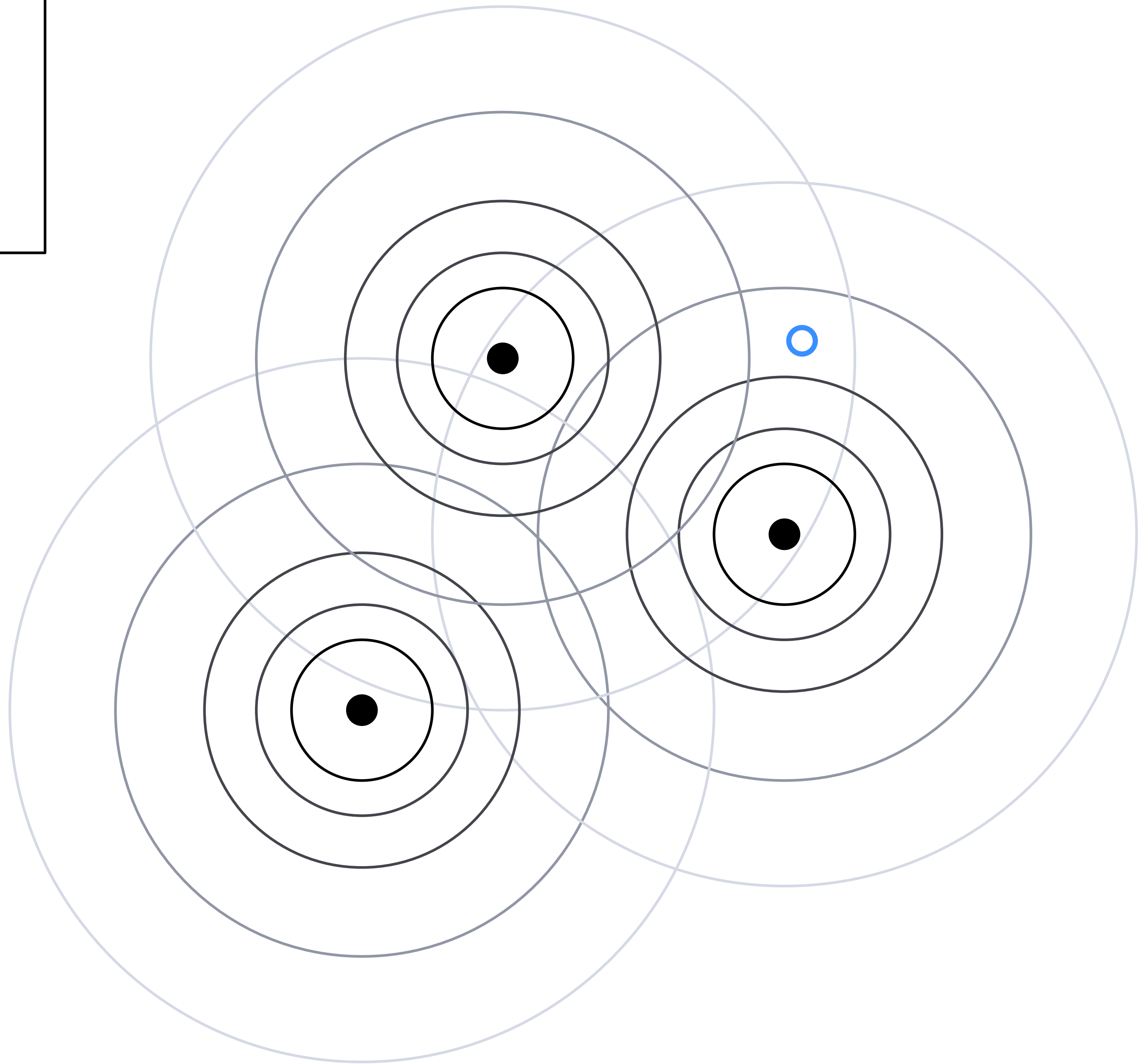
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



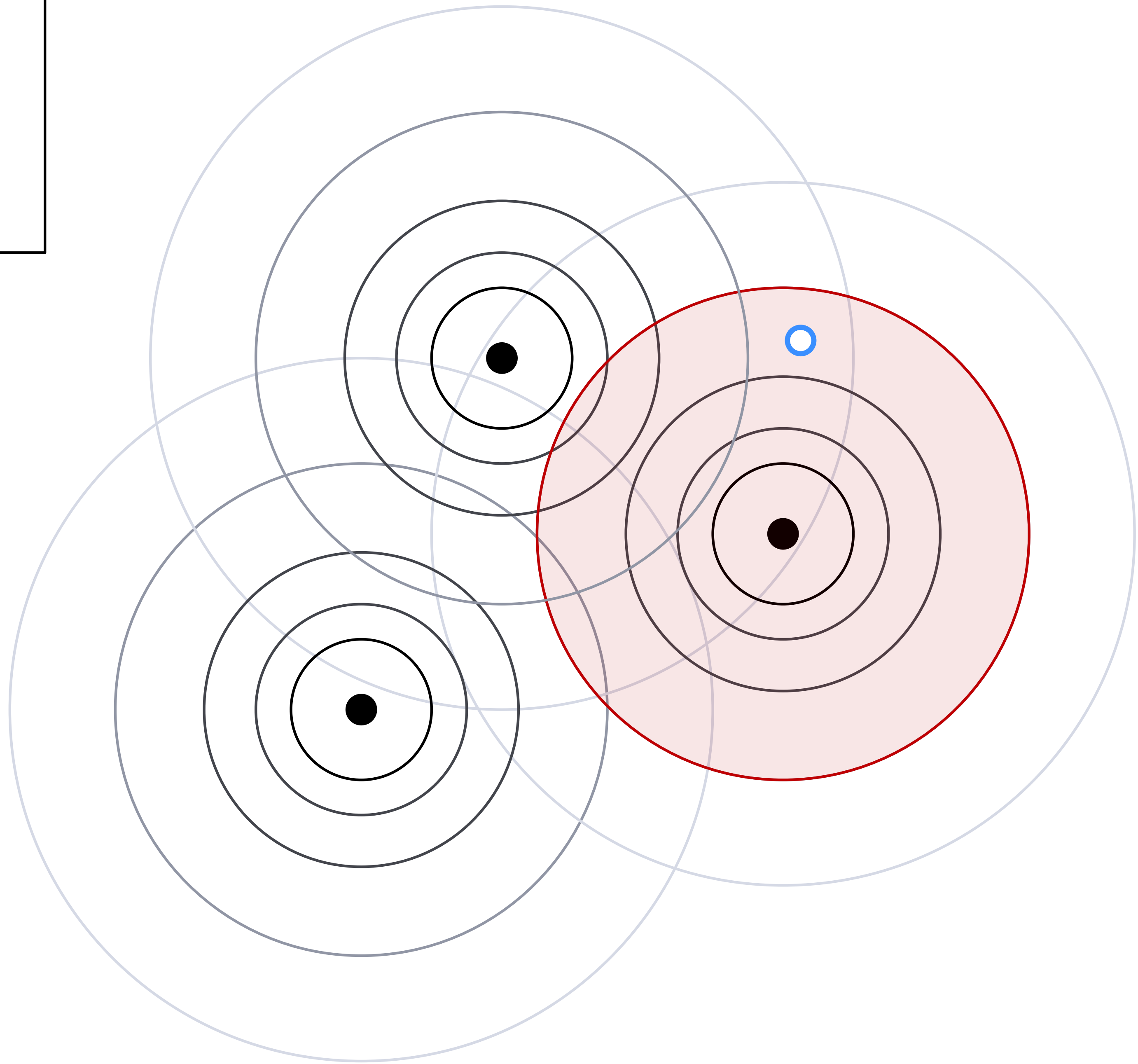
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



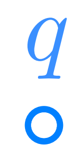
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



Reduction

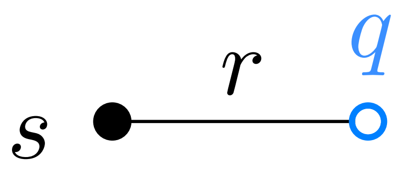
Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

Proof:
Let $s = nn(q, P)$ and $r = ||q - s|| = d(q, P)$.



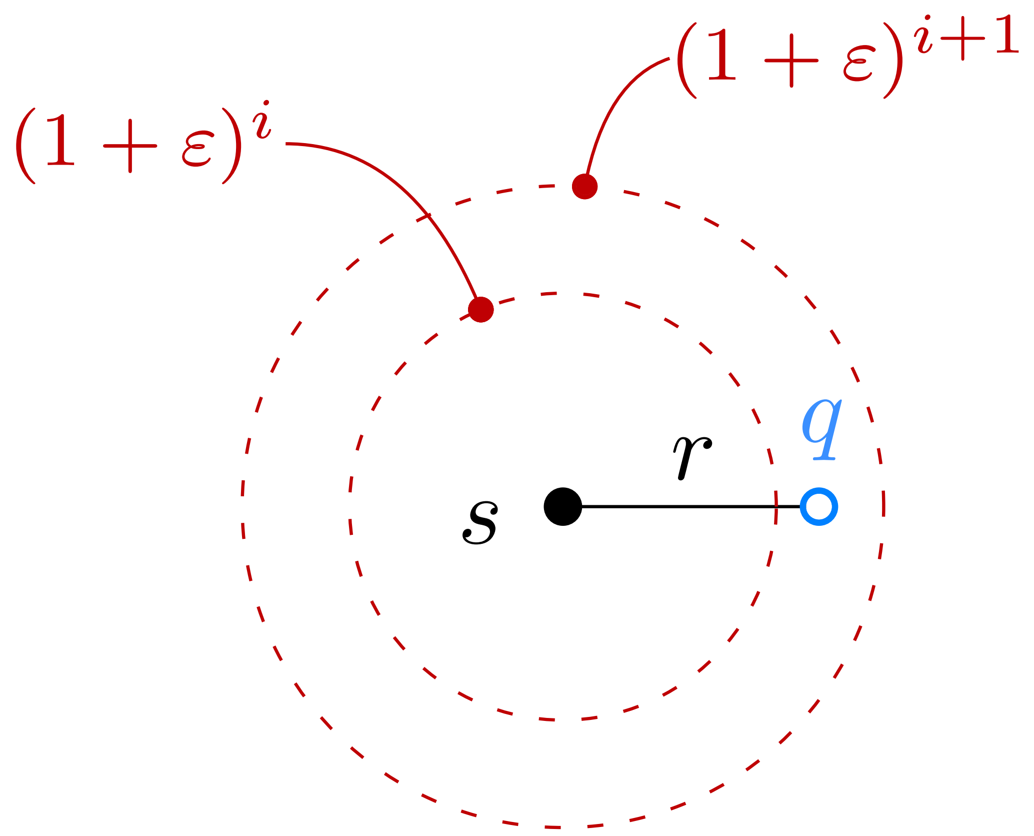
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

Proof:
Let $s = nn(q, P)$ and $r = ||q - s|| = d(q, P)$.

Pick i such that $(1+\varepsilon)^i < r \leq (1+\varepsilon)^{i+1}$.

We have that $q \in b(s, (1+\varepsilon)^{i+1})$



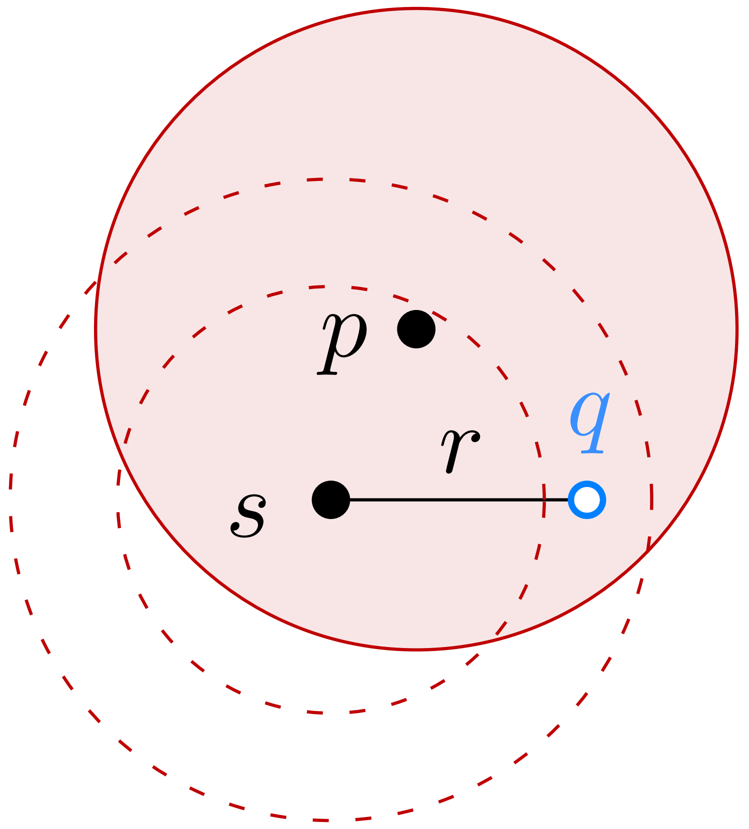
Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

Proof:
Let $s = nn(q, P)$ and $r = ||q - s|| = d(q, P)$.

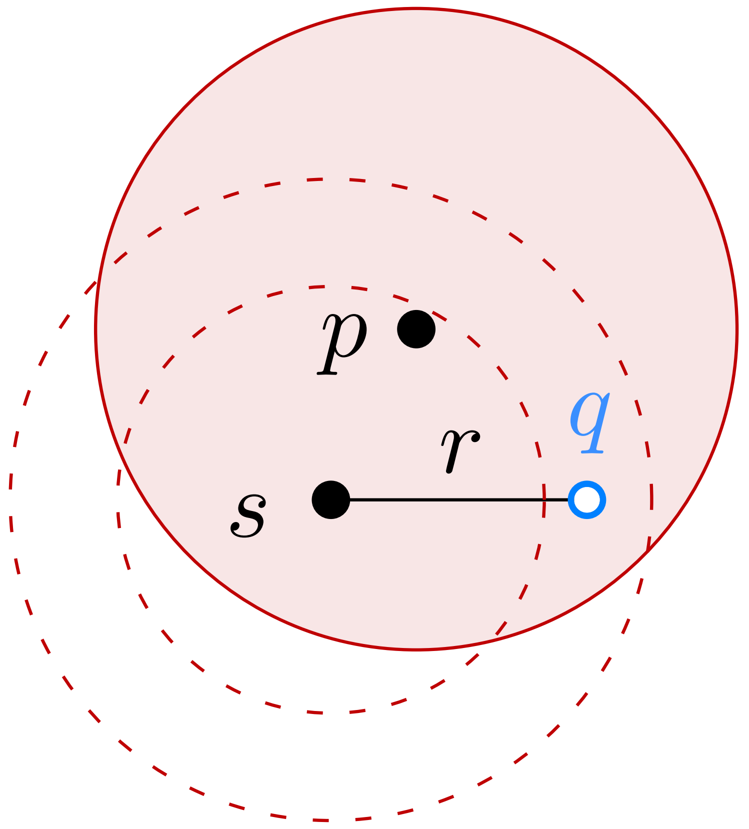
Pick i such that $(1+\varepsilon)^i < r \leq (1+\varepsilon)^{i+1}$.
We have that $q \in b(s, (1+\varepsilon)^{i+1})$

It must be that the target ball has a radius $\leq (1+\varepsilon)^{i+1}$.
It cannot be smaller than r, or bigger than $(1+\varepsilon)^{i+1}$.



Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



Proof:
Let $s = nn(q, P)$ and $r = ||q - s|| = d(q, P)$.

Pick i such that $(1+\varepsilon)^i < r \leq (1+\varepsilon)^{i+1}$.

We have that $q \in b(s, (1+\varepsilon)^{i+1})$

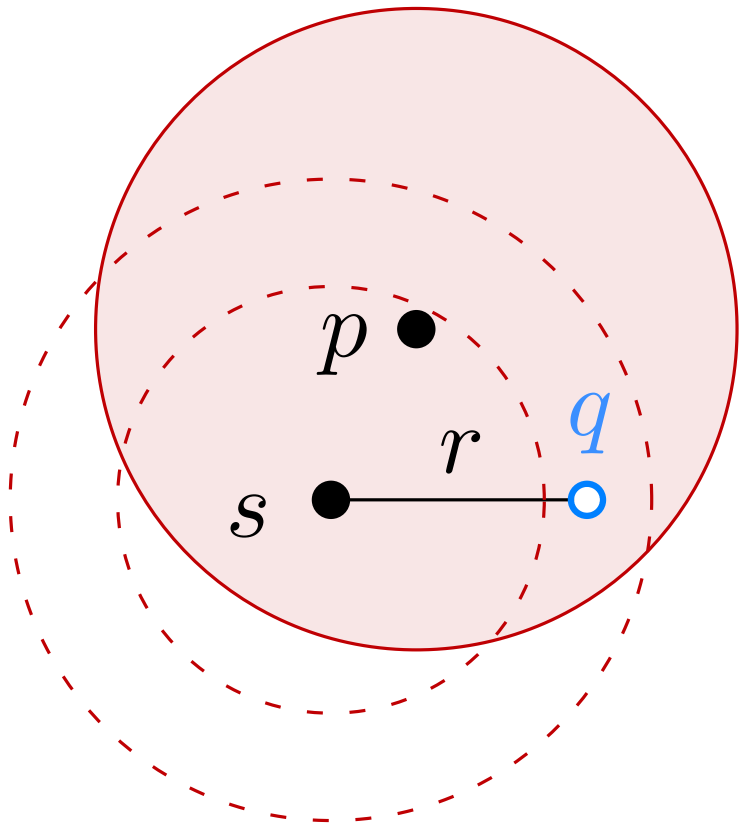
It must be that the target ball has a radius $\leq (1+\varepsilon)^{i+1}$.

It cannot be smaller than r, or bigger than $(1+\varepsilon)^{i+1}$.

It follows that $||q - p|| \leq radius(\odot_{\mathcal{B}}(q)) \leq (1+\varepsilon)^{i+1} < (1+\varepsilon)d(q, P)$

Reduction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



Proof:
Let $s = nn(q, P)$ and $r = ||q - s|| = d(q, P)$.

Pick i such that $(1+\varepsilon)^i < r \leq (1+\varepsilon)^{i+1}$.

We have that $q \in b(s, (1+\varepsilon)^{i+1})$

It must be that the target ball has a radius $\leq (1+\varepsilon)^{i+1}$.

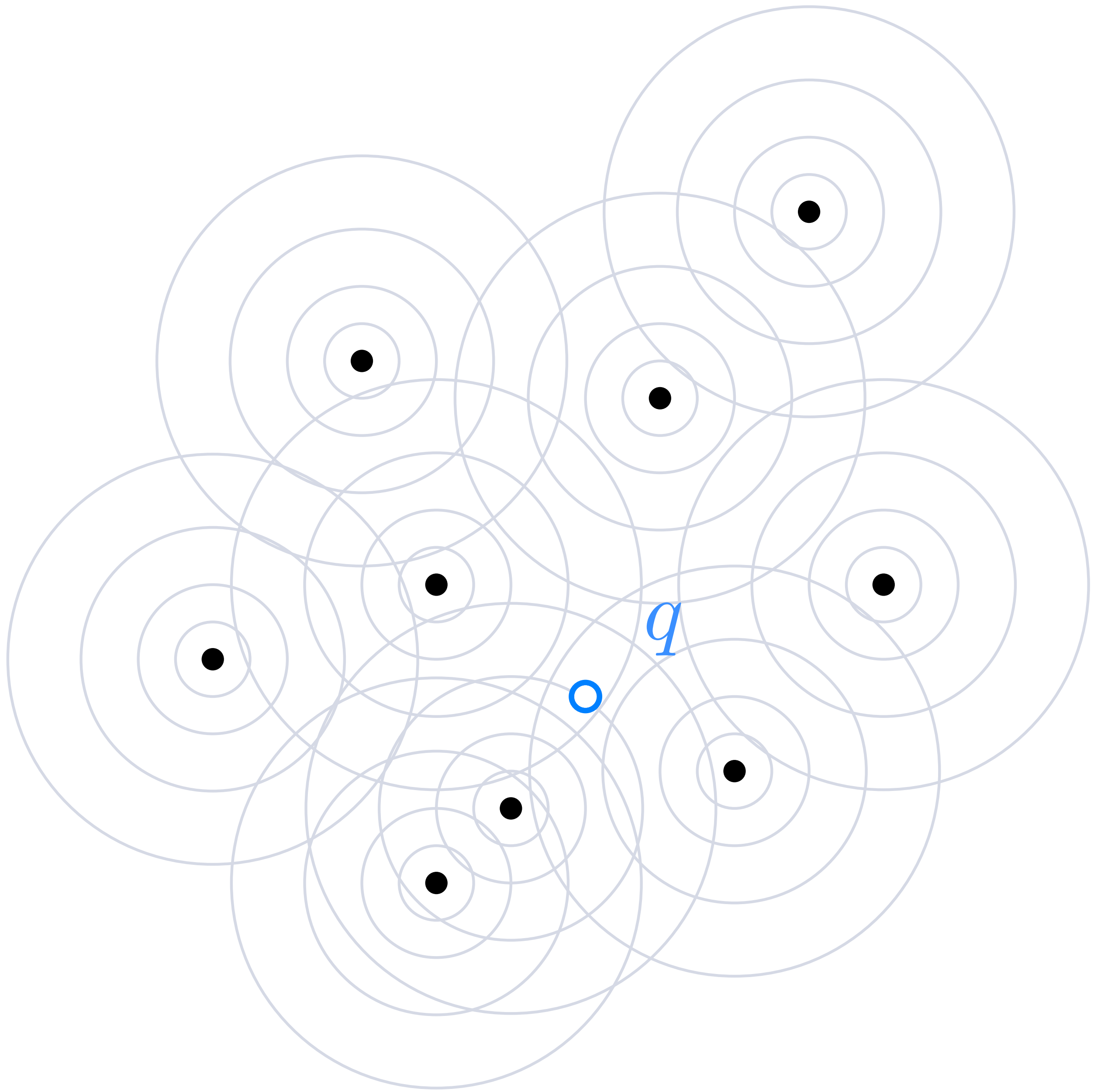
It cannot be smaller than r, or bigger than $(1+\varepsilon)^{i+1}$.

It follows that $||q - p|| \leq radius(\odot_{\mathcal{B}}(q)) \leq (1+\varepsilon)^{i+1} < (1+\varepsilon)d(q, P)$

p is $(1+\varepsilon)$-ANN to q!

First ideas – simple construction

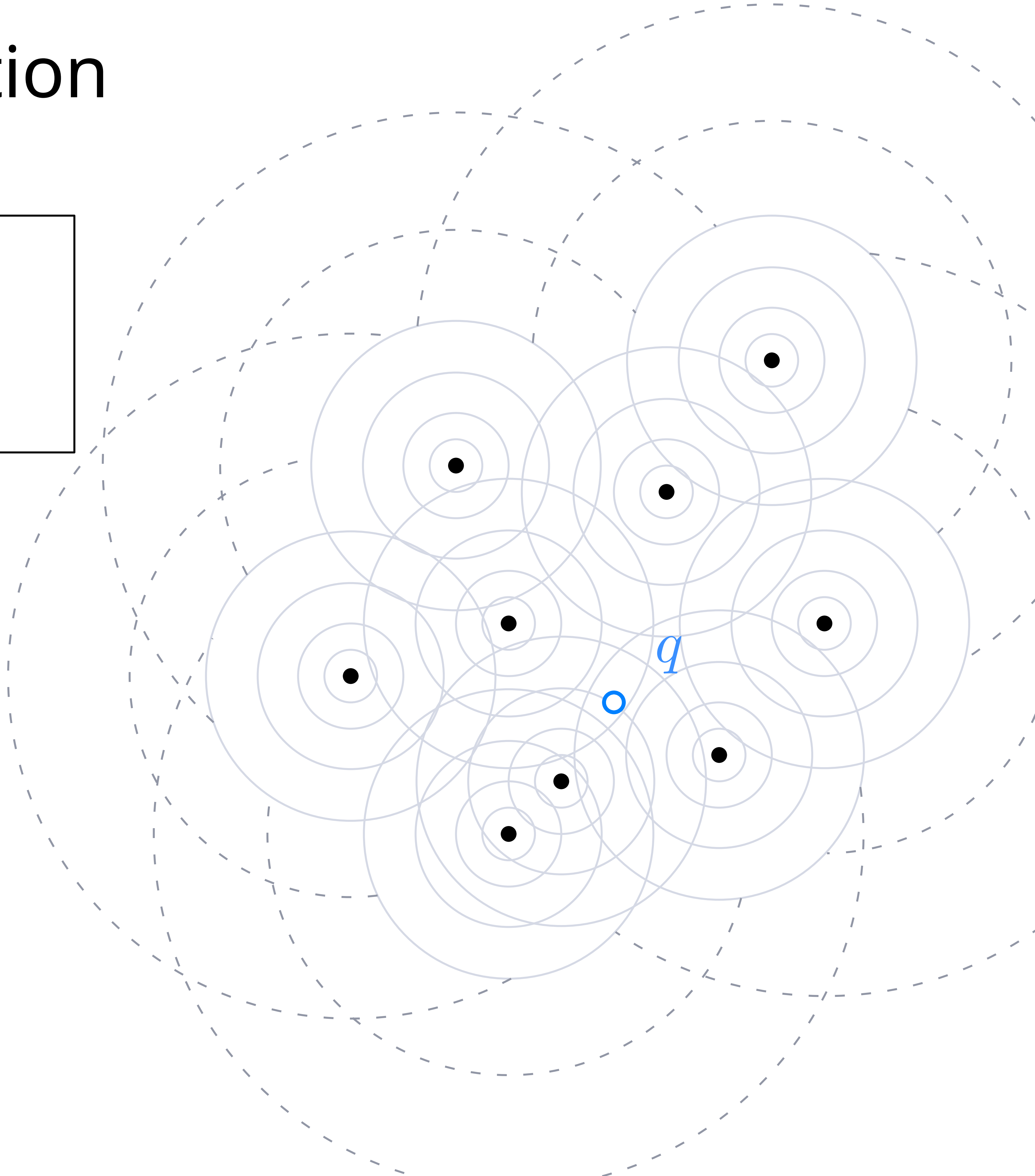
Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$*.*
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$*.*
Then p *is* $(1+\varepsilon)$*-ANN to* q*.*



First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

One issue.. ..$|\mathcal{B}|$ is unbounded!

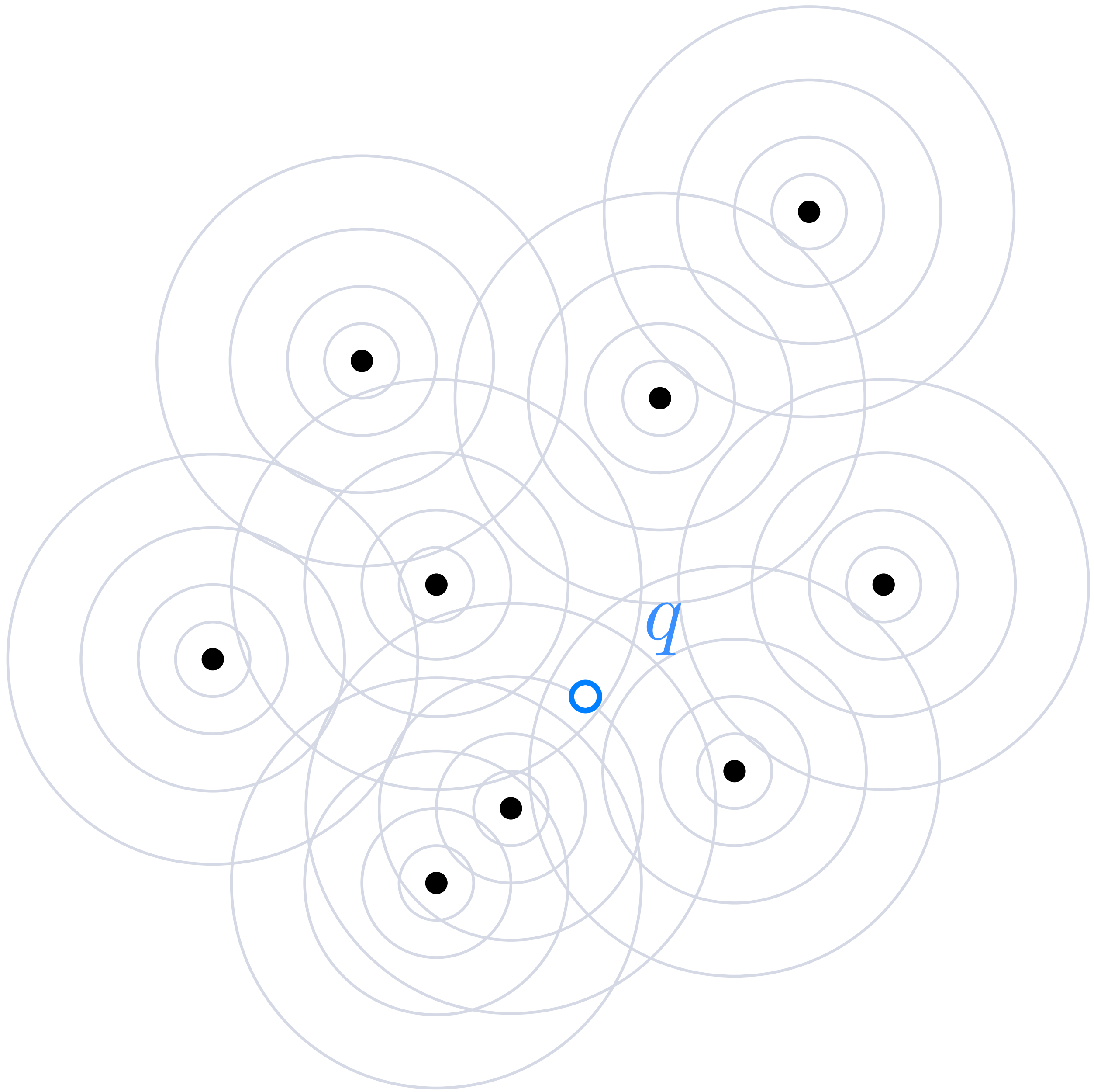


First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$*.*
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$*.*
Then p *is* $(1+\varepsilon)$*-ANN to* q*.*

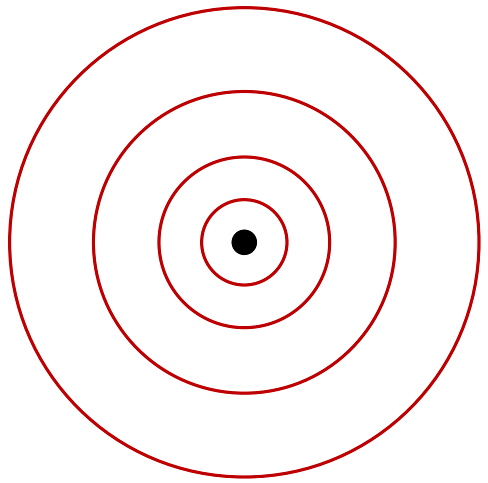
One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii



First ideas – simple construction

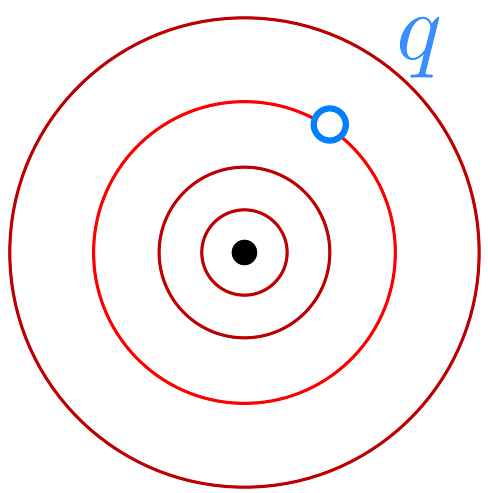
Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



One issue.. ..$|\mathcal{B}|$ is unbounded!

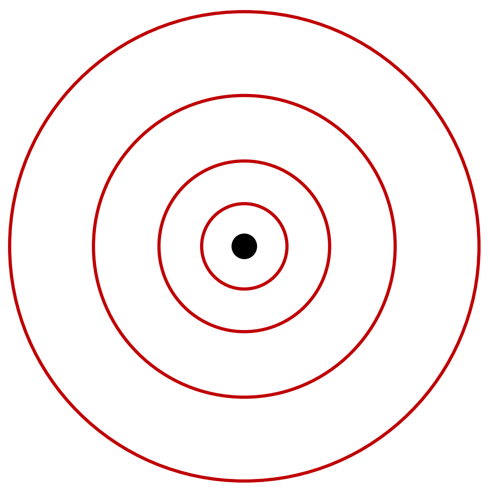
Solution: limit the range of radii

Consider pair of points $u, v \in P$



First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



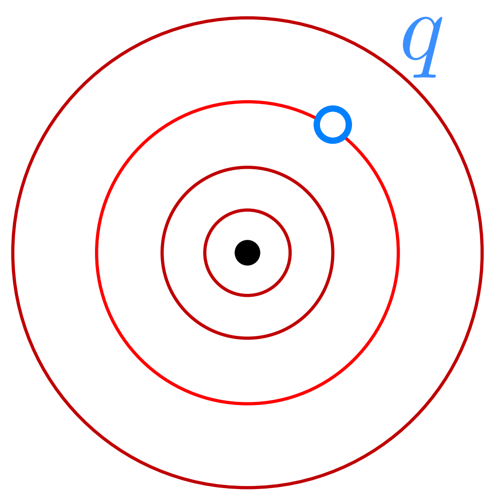
One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

Consider pair of points $u, v \in P$

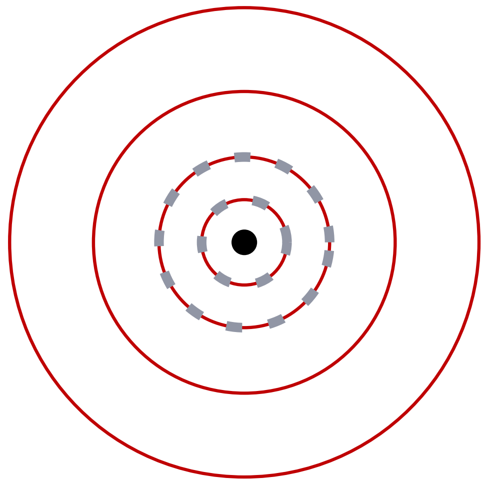
1. if q much closer to u than to v: easy to decide

   e.g. if $d(q,u) \leq d(u,v)/4$



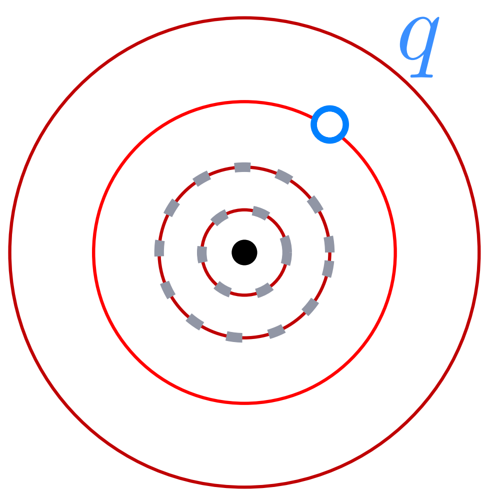
First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

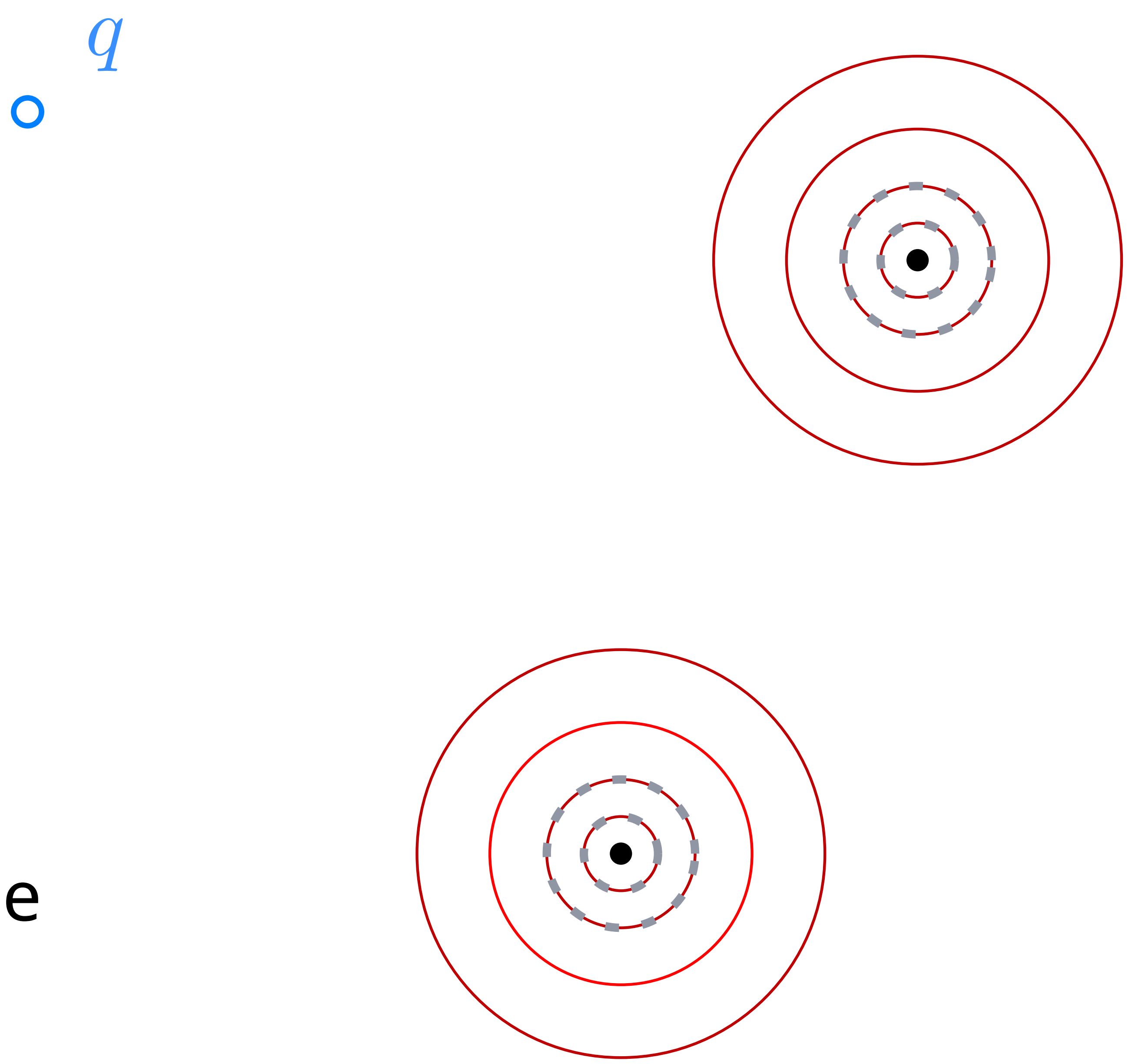


Consider pair of points $u, v \in P$

1. if q much closer to u than to v: easy to decide
   e.g. if $d(q,u) \leq d(u,v)/4$

First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.



One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

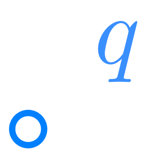
Consider pair of points $u, v \in P$

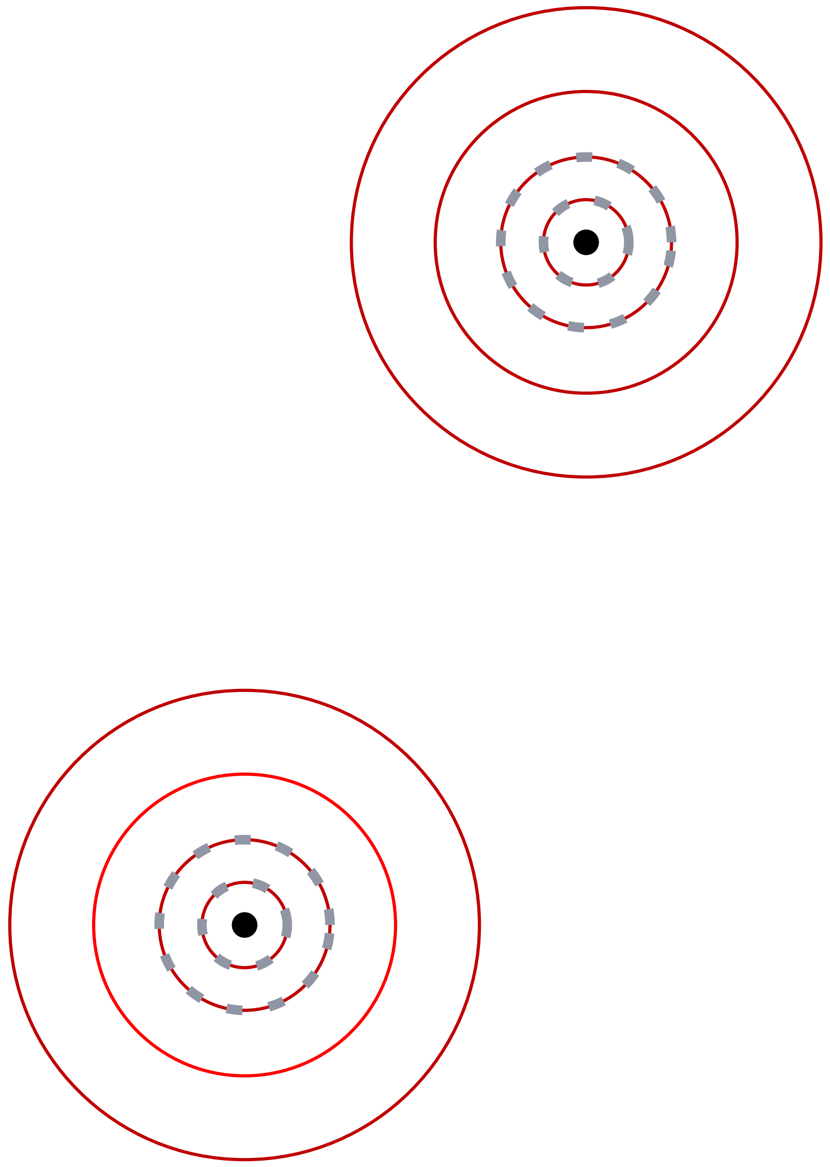
1. if q much closer to u than to v: easy to decide
   e.g. if $d(q,u) \leq d(u,v)/4$

2. if q very far from u and v: choice does not matter

First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.





One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

Consider pair of points $u, v \in P$

1. if q much closer to u than to v: easy to decide
   e.g. if $d(q,u) \leq d(u,v)/4$

2. if q very far from u and v: choice does not matter
   i.e. if $d(q,u) \geq 2d(u,v)/\varepsilon$

First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q, *let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

Observations:

(1) We only need range of radii:
$r/d(u,v) \in [1/4, 2/\varepsilon]$

One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

Consider pair of points $u, v \in P$

1. if q much closer to u than to v: easy to decide
   e.g. if $d(q,u) \leq d(u,v)/4$

2. if q very far from u and v: choice does not matter
   i.e. if $d(q,u) \geq 2d(u,v)/\varepsilon$

First ideas – simple construction

Lemma: *Let* $\mathcal{B} = \bigcup_{i=-\infty}^{\infty} \mathcal{U}(P, (1+\varepsilon)^i)$.
For a query q*, let* p *be the center of* $\odot_{\mathcal{B}}(q)$.
Then p *is* $(1+\varepsilon)$*-ANN to* q.

One issue.. ..$|\mathcal{B}|$ is unbounded!

Solution: limit the range of radii

Consider pair of points $u, v \in P$

1. if q much closer to u than to v: easy to decide
   e.g. if $d(q,u) \leq d(u,v)/4$

2. if q very far from u and v: choice does not matter
   i.e. if $d(q,u) \geq 2d(u,v)/\varepsilon$

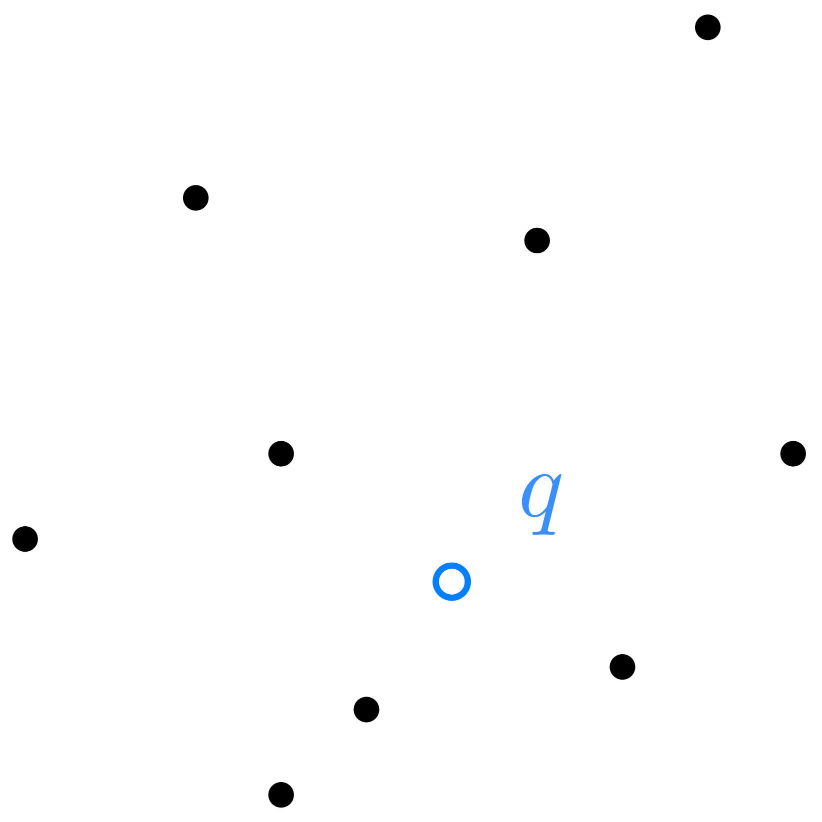
Observations:

(1) We only need range of radii:
$r/d(u,v) \in [1/4, 2/\varepsilon]$

(2) need to avoid range dependent on pairs, otherwise $\Theta(n^2)$ disks

Handling a range of radii

Near neighbor data structure $\mathcal{D}(P, r)$



Handling a range of radii

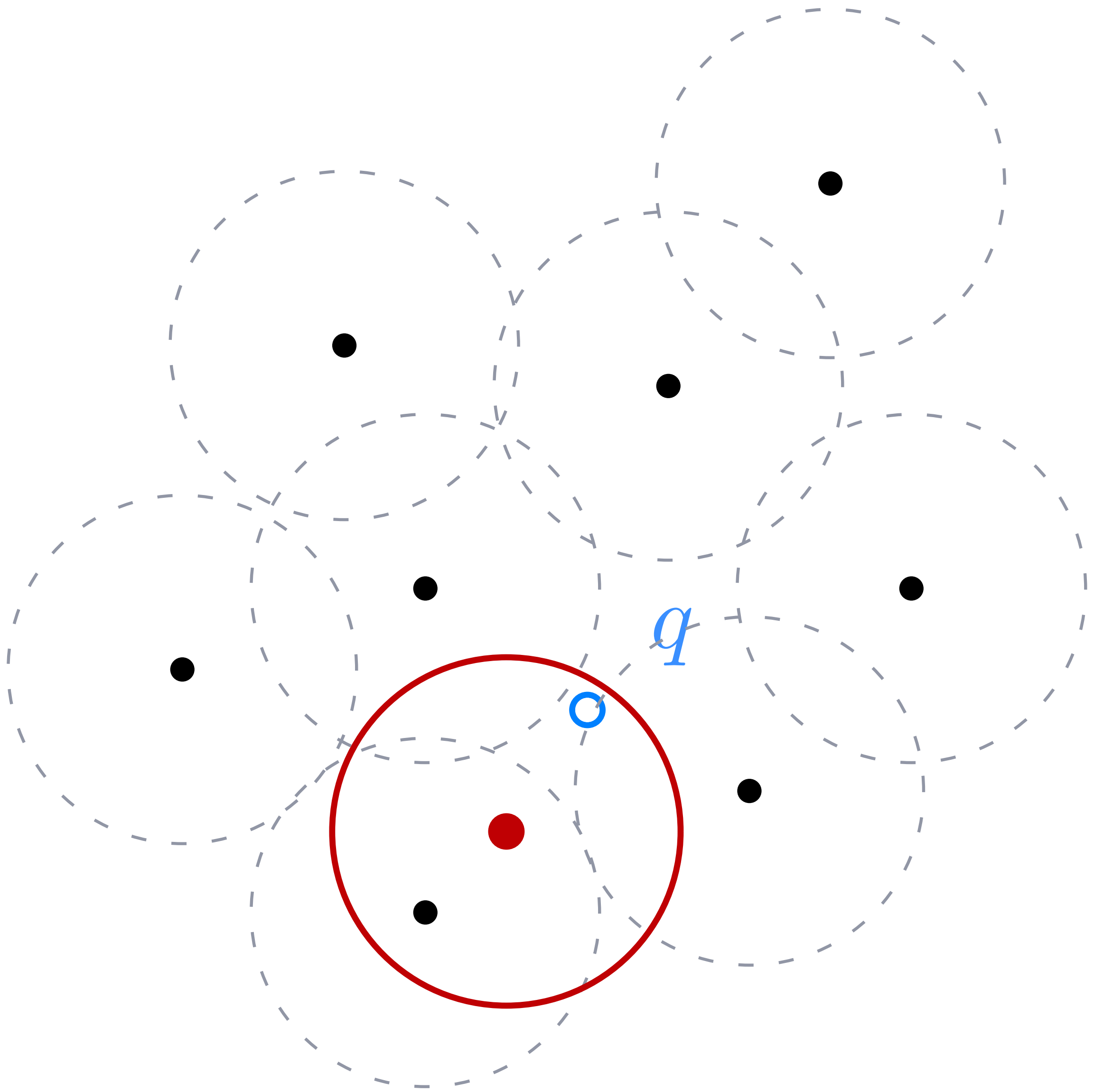
Near neighbor data structure $\mathcal{D}(P, r)$

Decides given q, if

$d(q, P) \leq r$, or

$d(q, P) > r$

If $d(q, P) \leq r$ it returns a point p s.t. $d(q, p) \leq r$



Handling a range of radii

Near neighbor data structure $\mathcal{D}(P, r)$

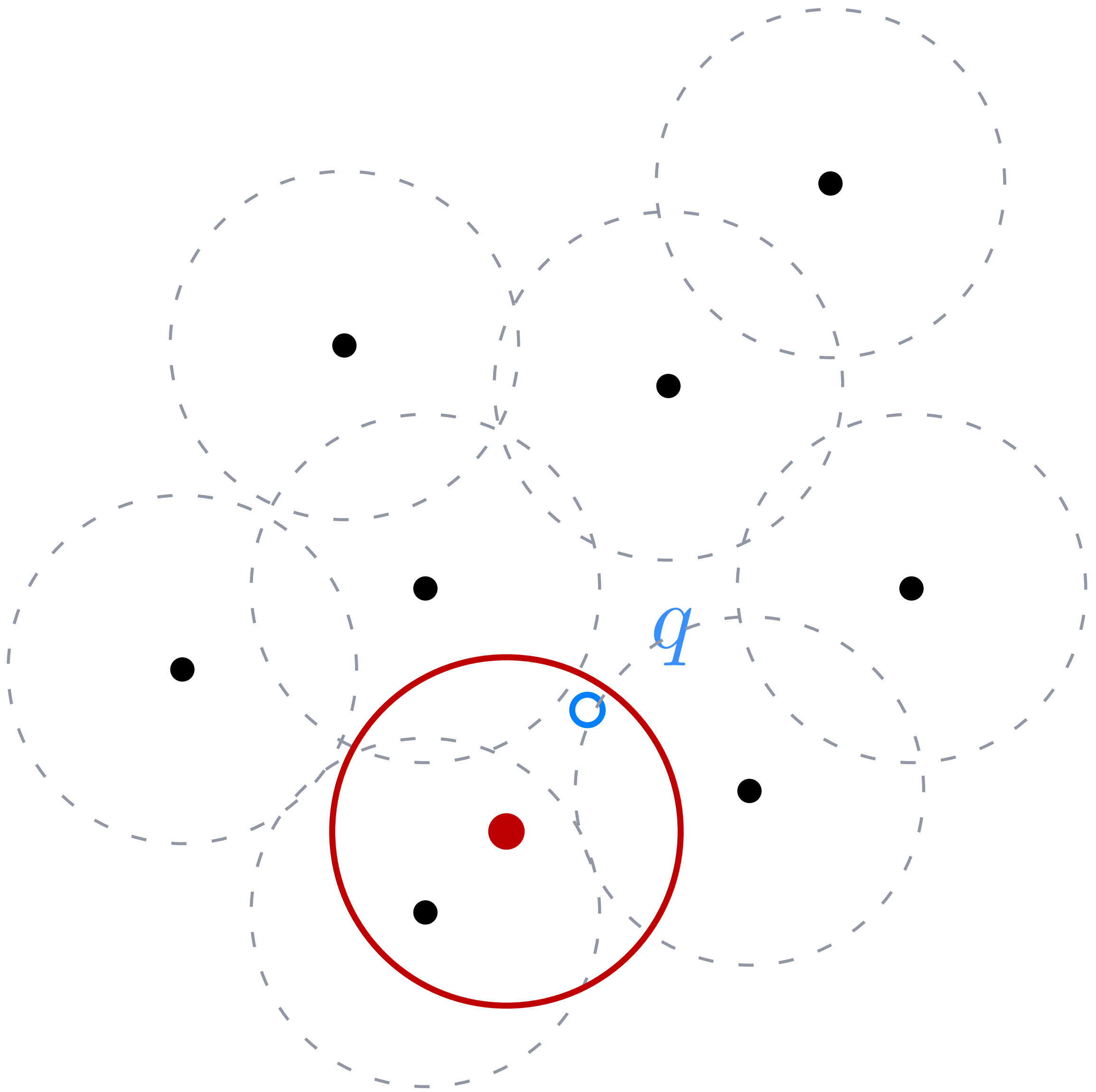
Decides given q, if

$d(q, P) \leq r$, or

$d(q, P) > r$

If $d(q, P) \leq r$ it returns a point p s.t. $d(q, p) \leq r$

A query can be resolved by iteratively checking for each ball in $\mathcal{U}(P, r)$ if it contains q

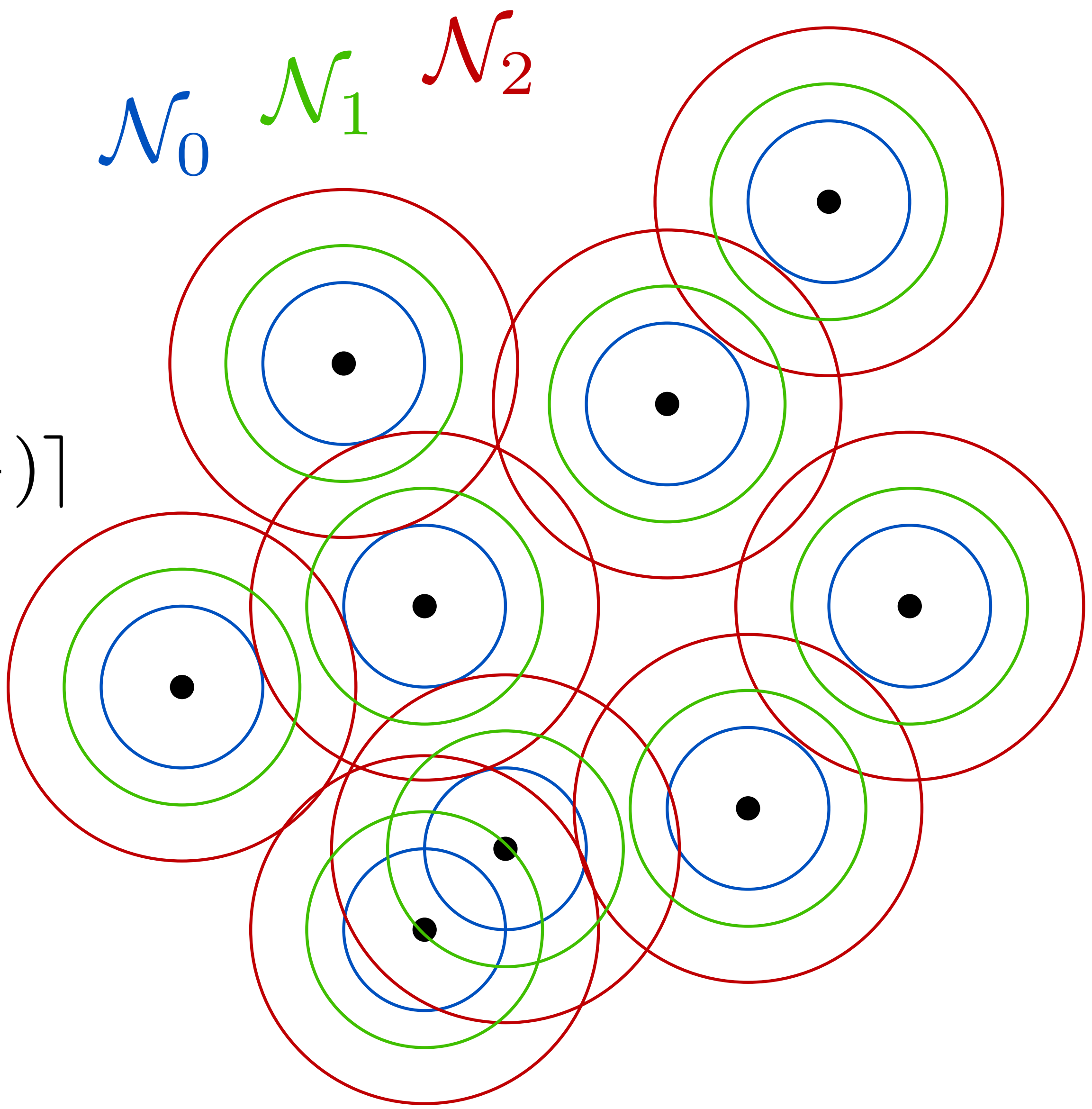


Handling a range of radii

Near neighbor data structure $\mathcal{D}(P, r)$

Given interval $[a, b]$, Let $\mathcal{N}_i = \mathcal{D}(P, r_i)$ where

$r_i = \min\left((1+\varepsilon)^i a, b\right)$ for $i = 0, \ldots, M = \lceil \log_{1+\varepsilon}(\frac{b}{a}) \rceil$

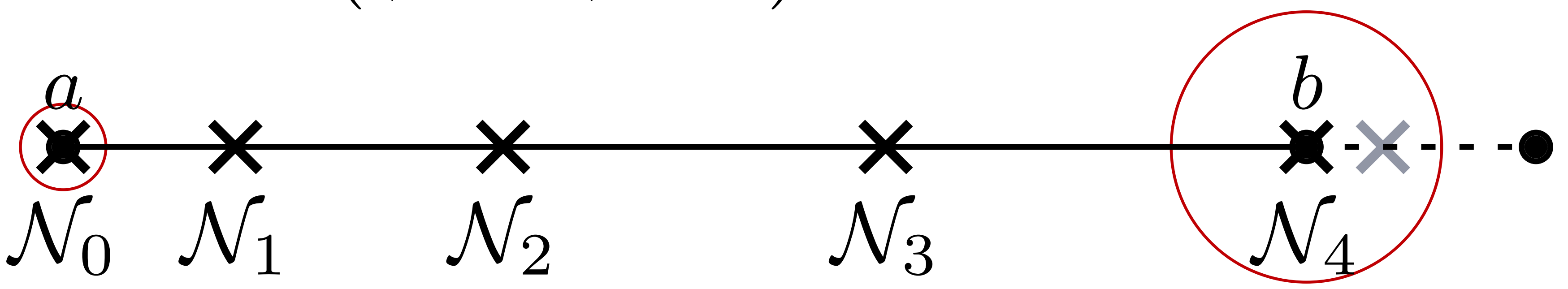


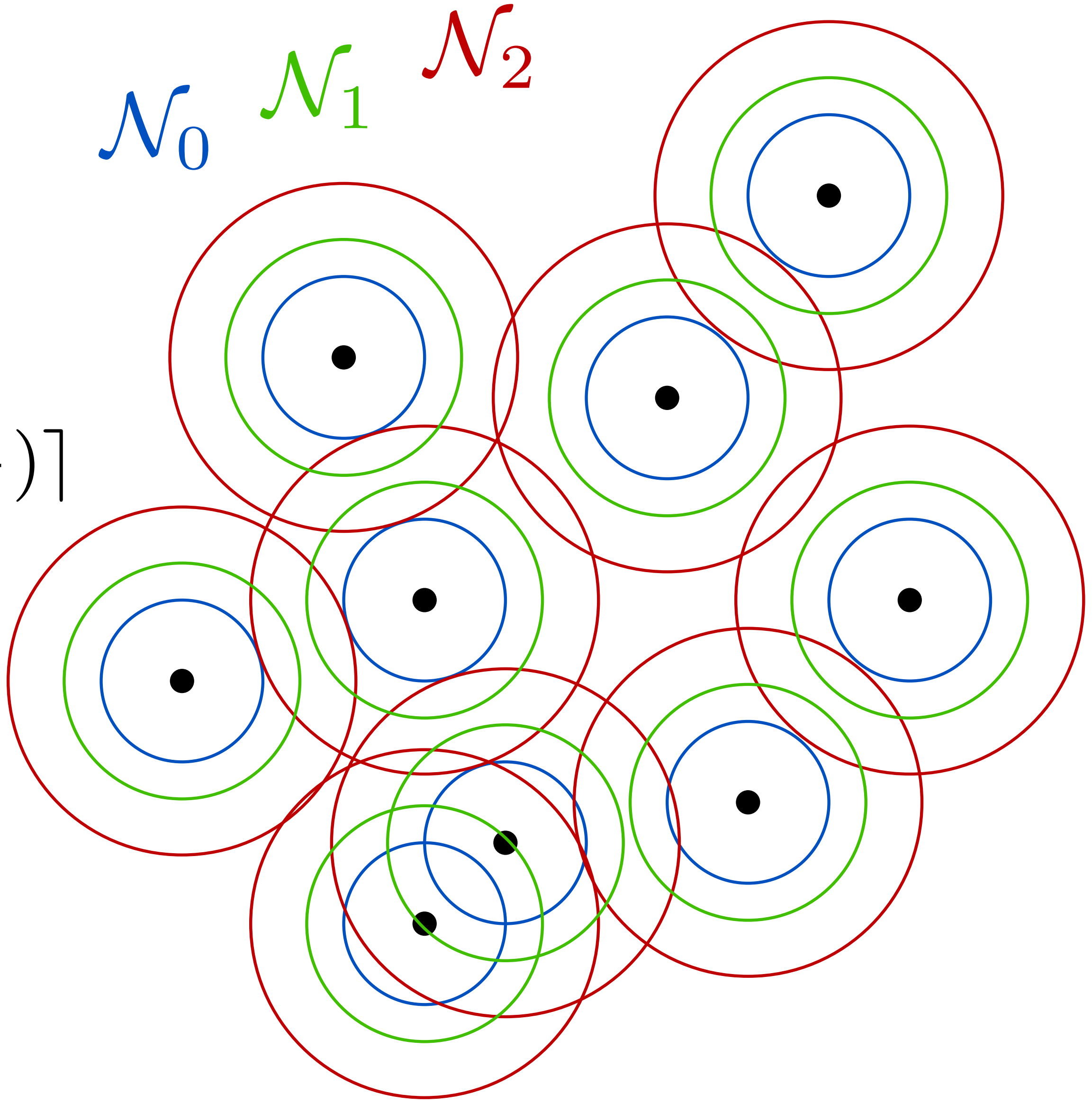
Handling a range of radii

Near neighbor data structure $\mathcal{D}(P, r)$

Given interval $[a, b]$, Let $\mathcal{N}_i = \mathcal{D}(P, r_i)$ where

$r_i = \min\left((1+\varepsilon)^i a, b\right)$ for $i = 0, \ldots, M = \lceil \log_{1+\varepsilon}(\frac{b}{a}) \rceil$



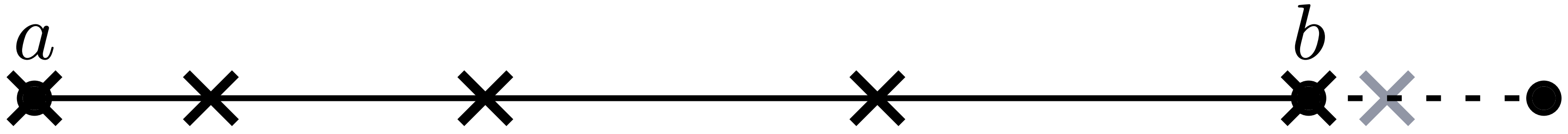


Handling a range of radii

Near neighbor data structure $\mathcal{D}(P, r)$

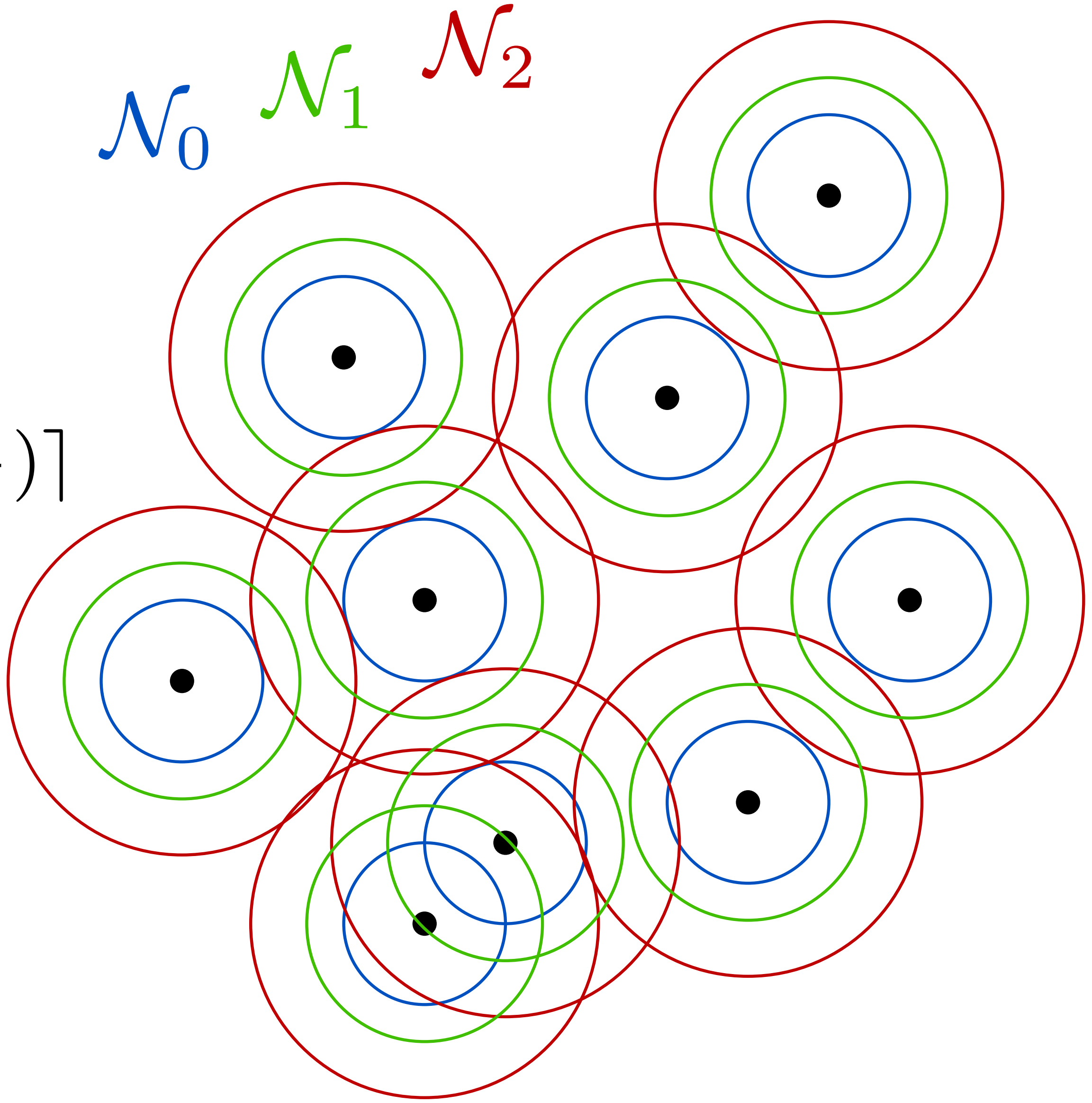
Given interval $[a, b]$, Let $\mathcal{N}_i = \mathcal{D}(P, r_i)$ where

$r_i = \min\left((1+\varepsilon)^i a, b\right)$ for $i = 0, \ldots, M = \lceil \log_{1+\varepsilon}(\frac{b}{a}) \rceil$



Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

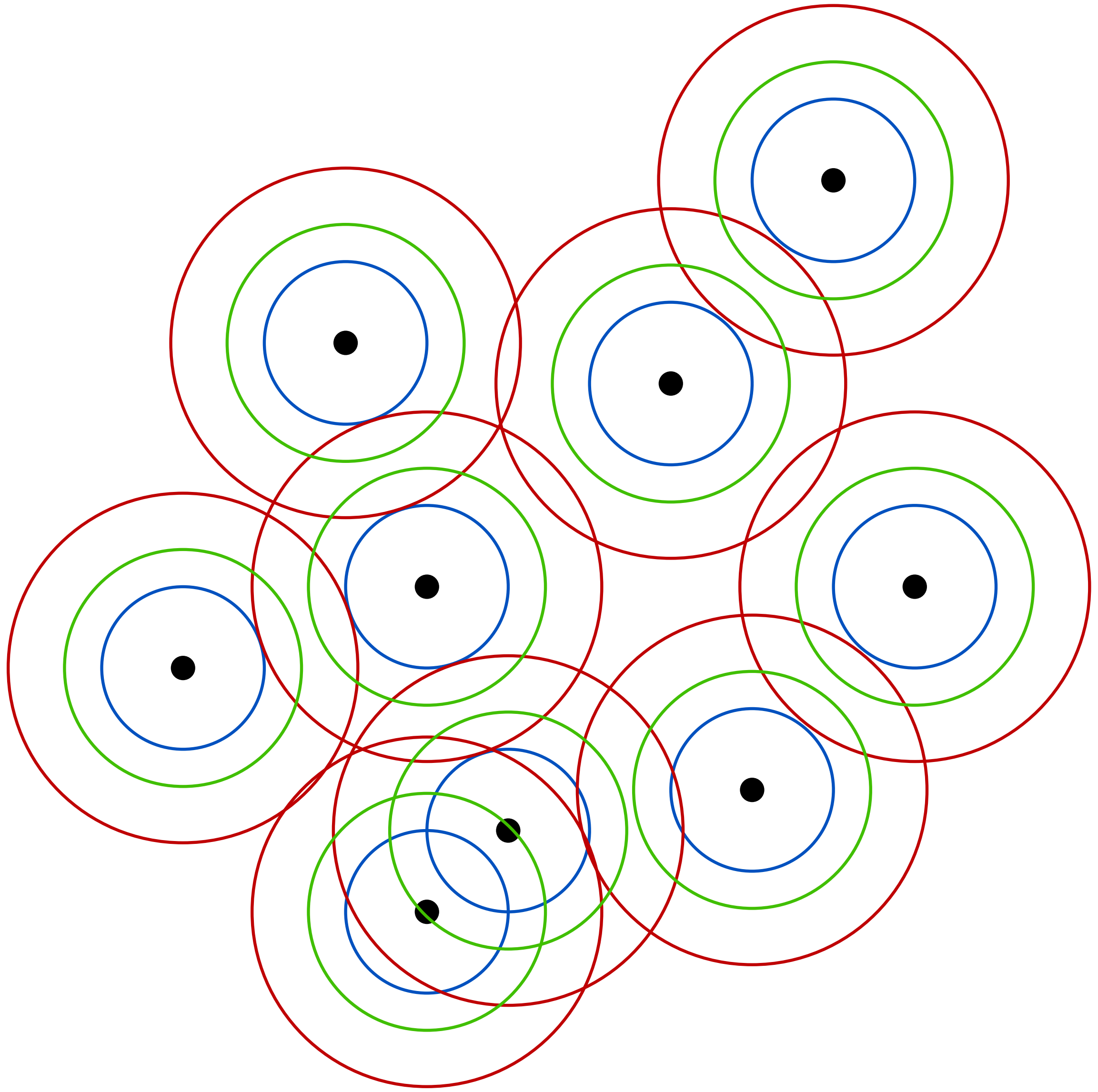
Let $\hat{\mathcal{I}}(P, a, b, \varepsilon) = \{\mathcal{N}_0, \ldots, \mathcal{N}_M\}$



Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

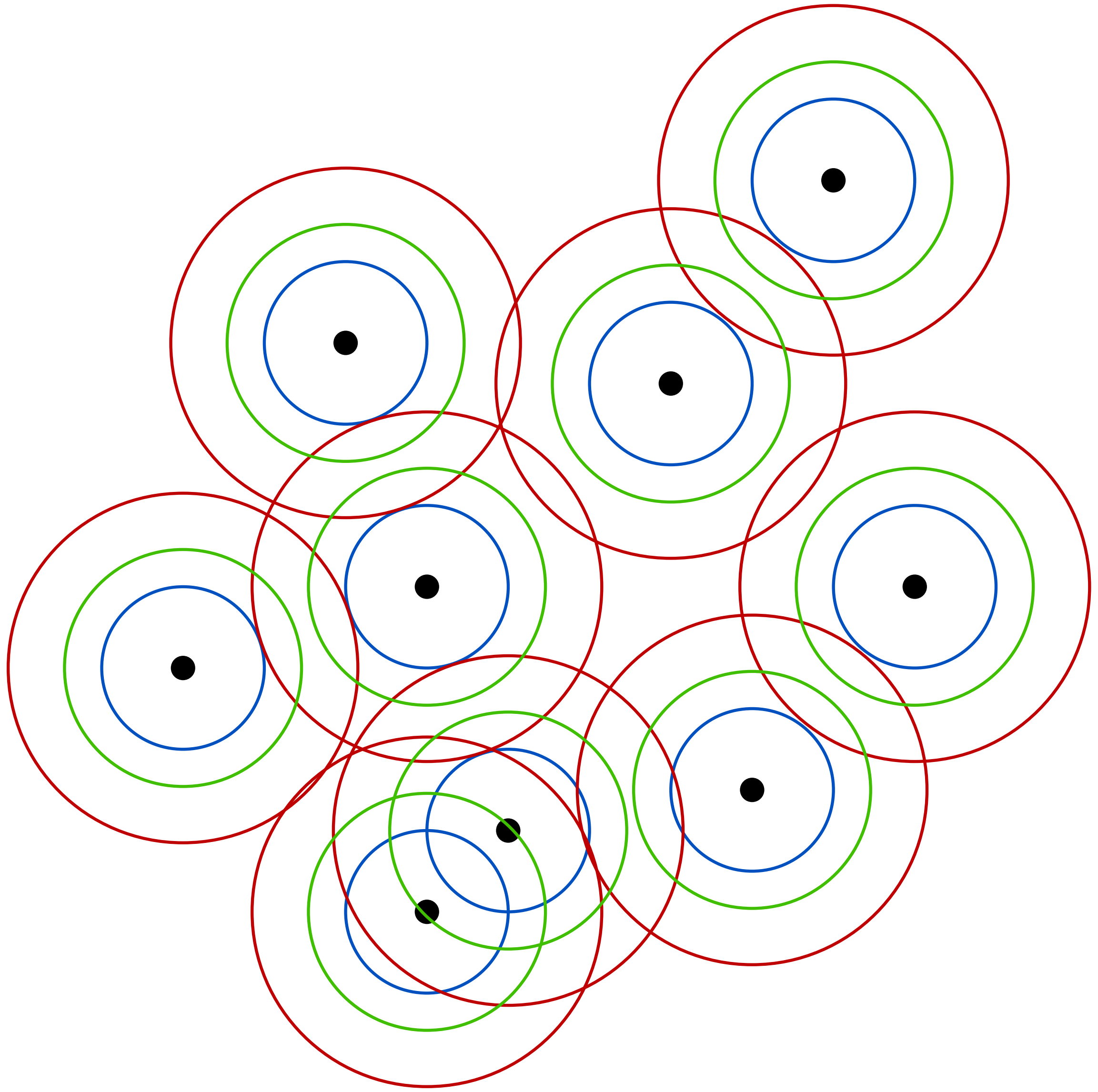


Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

1. $d(q, P) \leq a$, *or*
2. $d(q, P) > b$ *and otherwise*
3. *Return a radius* r *and point* p *s.t.*
   $d(q, P) \leq r = d(q, p) \leq (1 + \varepsilon) d(q, P)$



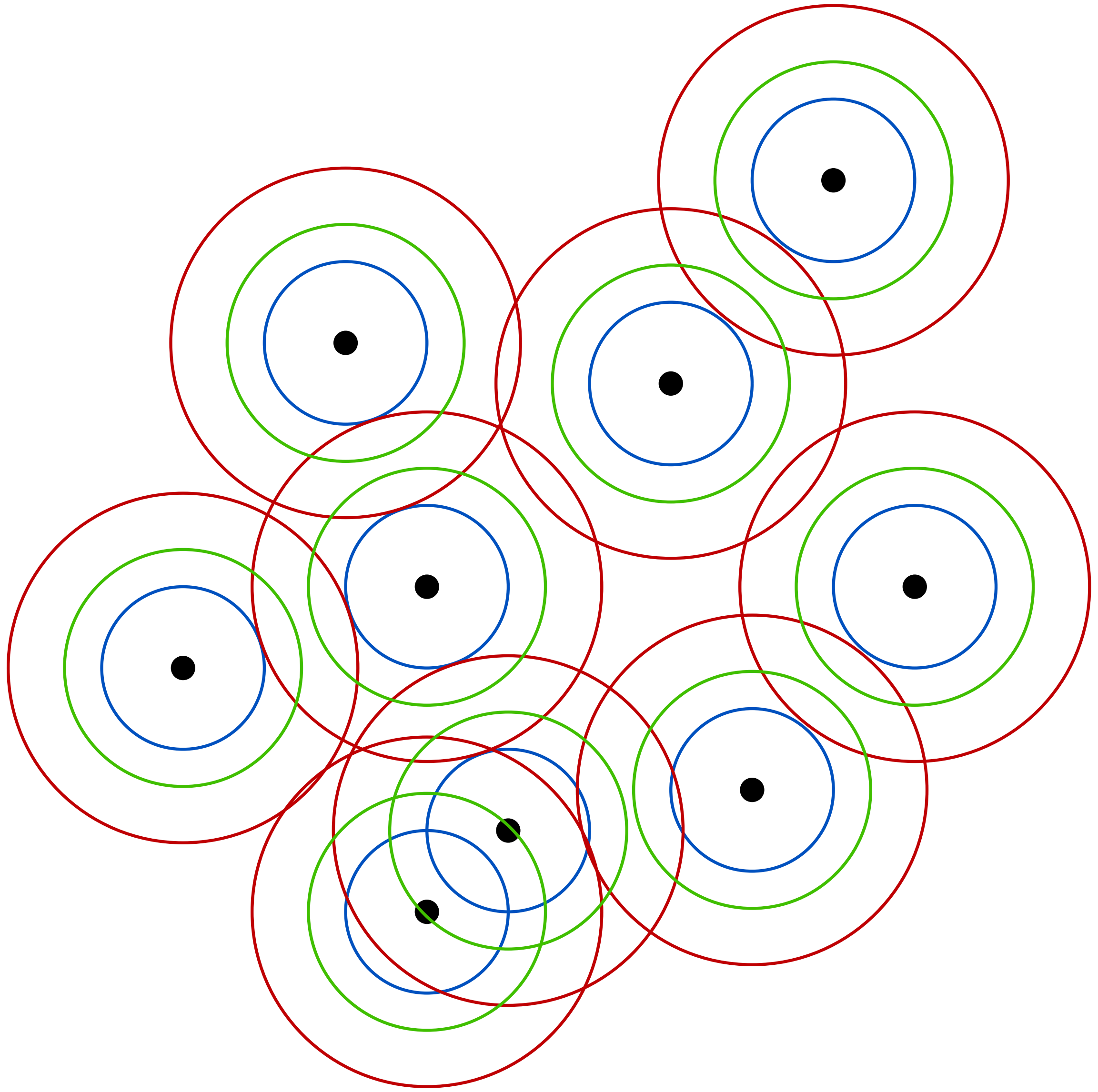
Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

1. $d(q, P) \leq a$, *or*
2. $d(q, P) > b$ *and otherwise*
3. *Return a radius* r *and point* p *s.t.*
   $d(q, P) \leq r = d(q, p) \leq (1 + \varepsilon) d(q, P)$

number of queries required is $O(\log(\varepsilon^{-1} \log(b/a)))$



Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

Recall, $M = \lceil \log_{1+\varepsilon}(b/a) \rceil$

Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

1. $d(q, P) \leq a$*, or*
2. $d(q, P) > b$ *and otherwise*
3. *Return a radius* r *and point* p *s.t.*

$$d(q, P) \leq r = d(q, p) \leq (1 + \varepsilon) d(q, P)$$

number of queries required is $O(\log(\varepsilon^{-1} \log(b/a)))$

Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

1. $d(q, P) \leq a$, *or*
2. $d(q, P) > b$ *and otherwise*
3. *Return a radius* r *and point* p *s.t.*

$$d(q, P) \leq r = d(q, p) \leq (1 + \varepsilon) d(q, P)$$

number of queries required is $O(\log(\varepsilon^{-1} \log(b/a)))$

Recall, $M = \lceil \log_{1+\varepsilon}(b/a) \rceil$

$O(\log(b/a)/\varepsilon)$ nn structures

Handling a range of radii

Interval near neighbor data structure $\hat{\mathcal{I}}(P, a, b, \varepsilon)$

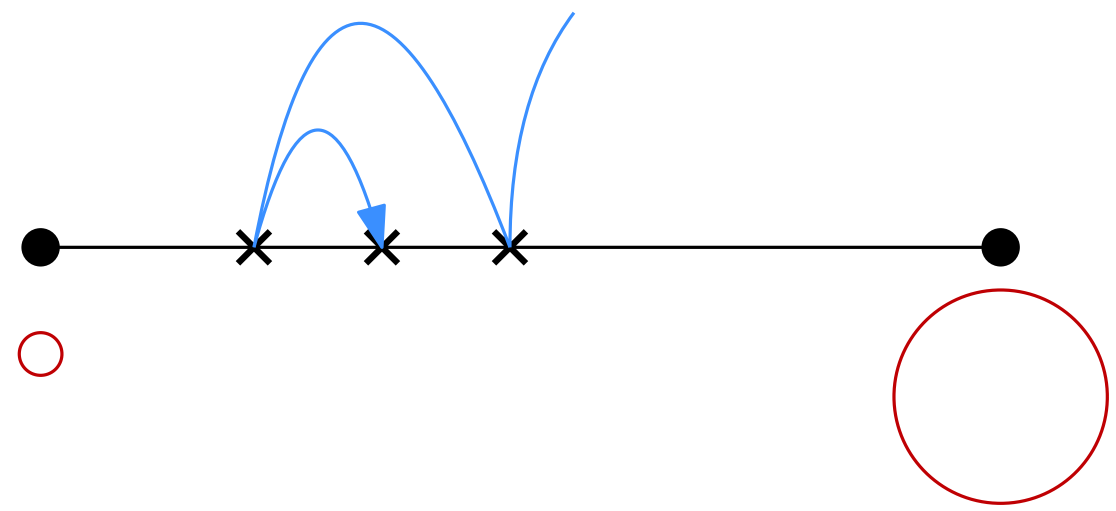
Lemma: Given P, $a \leq b$ and $\varepsilon > 0$, one can construct $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ such that: (A) $\hat{\mathcal{I}}$ is made out of $O(\varepsilon^{-1} \log(b/a))$ nn structures, and (B) given a query point q it can decide if:

1. $d(q, P) \leq a$, *or*
2. $d(q, P) > b$ *and otherwise*
3. *Return a radius* r *and point* p *s.t.*
   $d(q, P) \leq r = d(q, p) \leq (1 + \varepsilon) d(q, P)$

number of queries required is $O(\log(\varepsilon^{-1} \log(b/a)))$

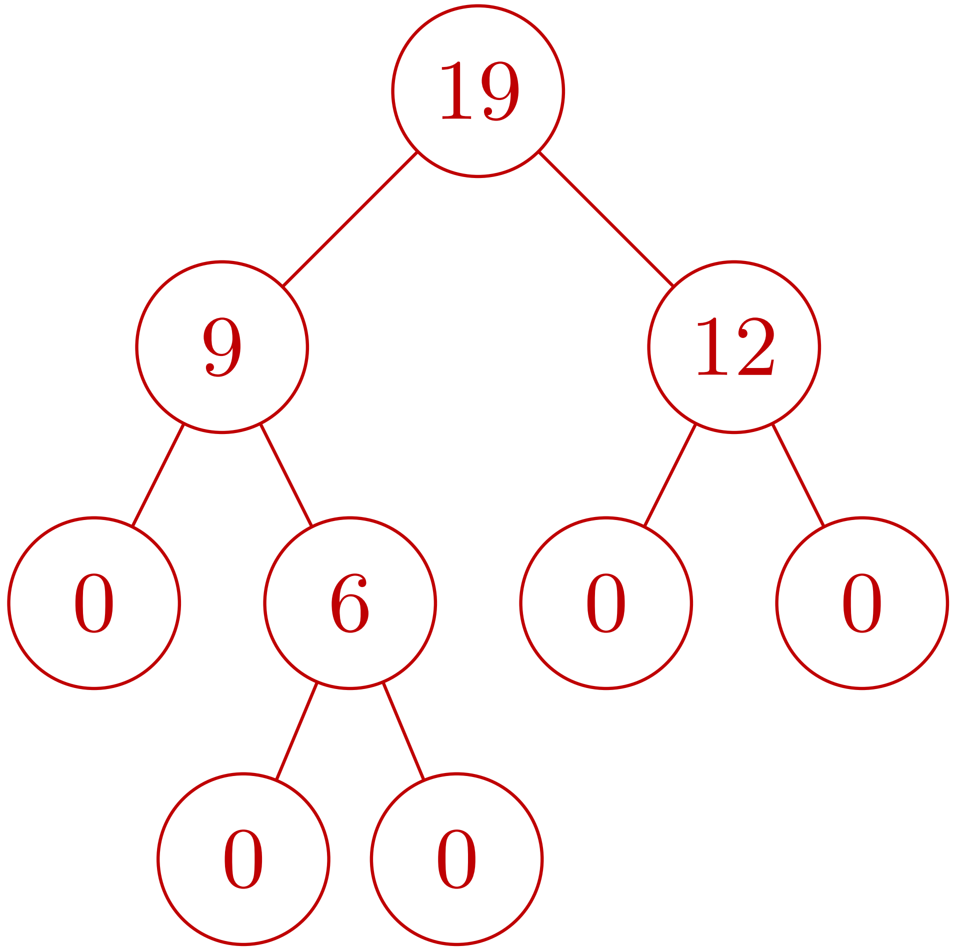
Recall, $M = \lceil \log_{1+\varepsilon}(b/a) \rceil$
$O(\log(b/a)/\varepsilon)$ nn structures

number of queries can be achieved by doing a binary search on the radius:
$O(\log(\varepsilon^{-1} \log(b/a)))$



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

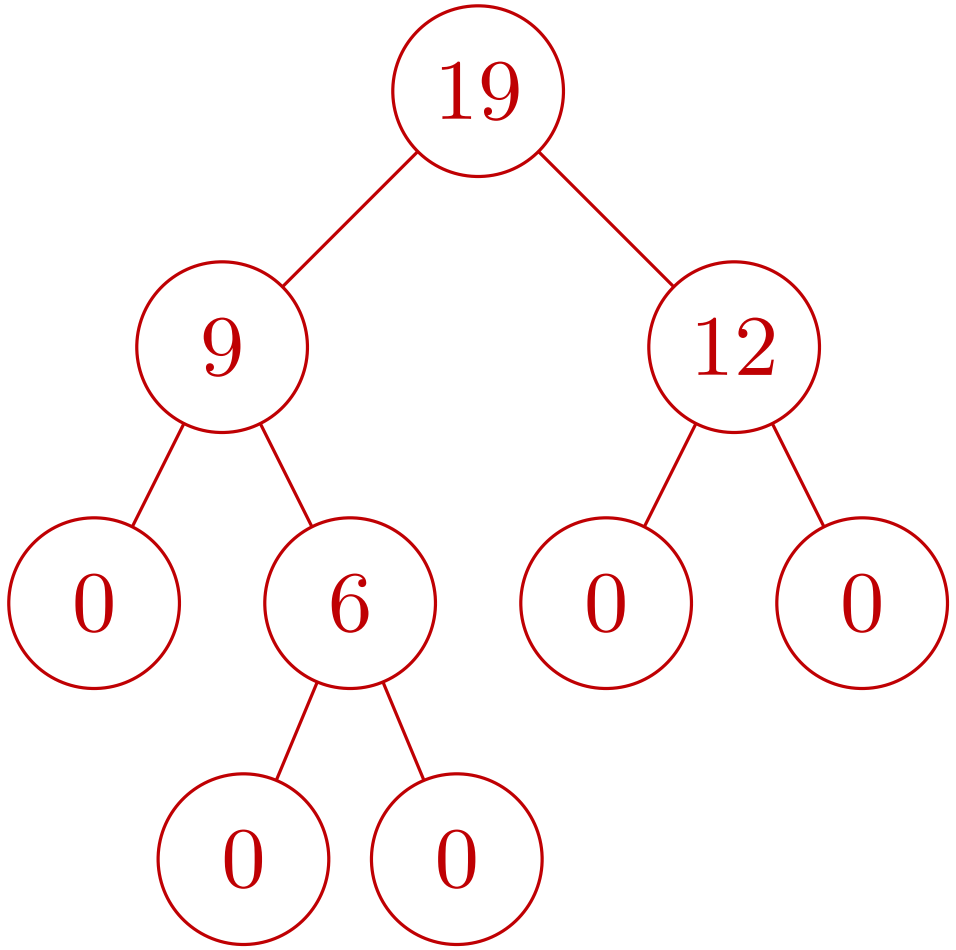


The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

recall:

- Each vertex v has a label $\Delta_v \geq 0$.
- $\Delta_v = 0$ if v is a leaf.
- If u is a child of v, then $\Delta_v \geq \Delta_u$
- $\Delta_{lca(u,v)}$ denotes the t-approximated distance between two leaves u and v



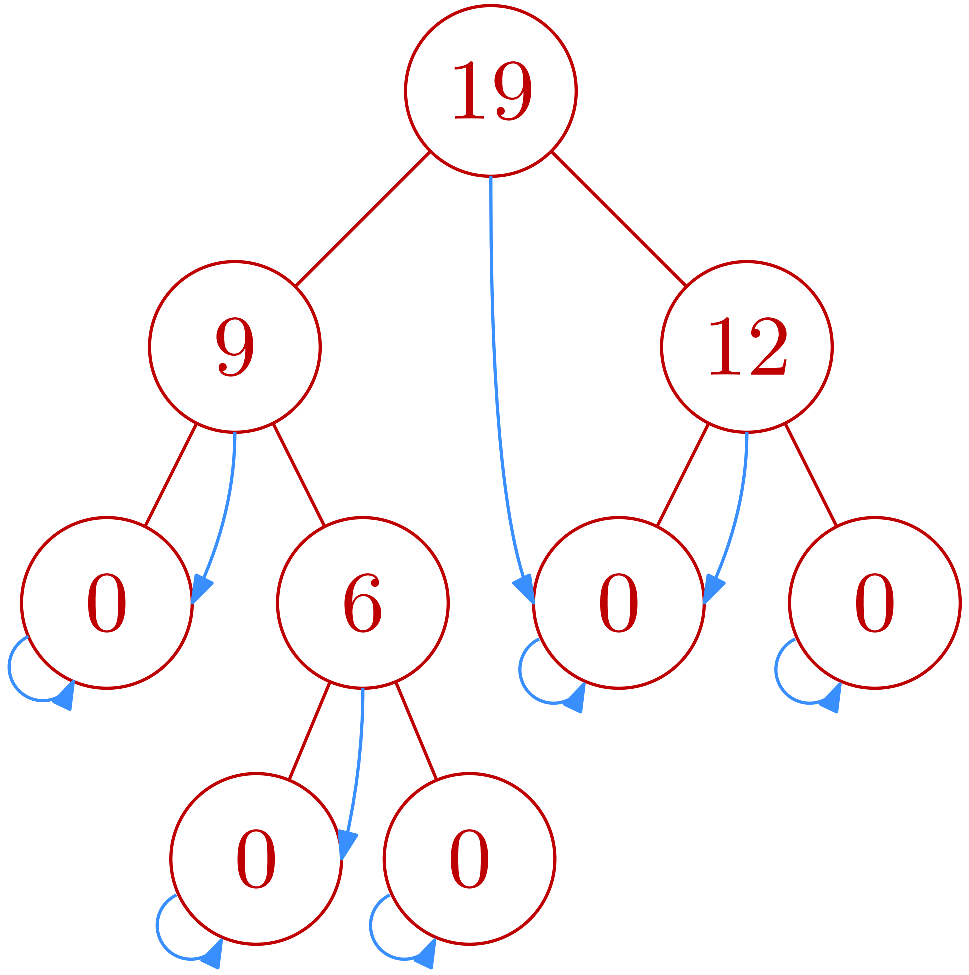
The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

recall:

- Each vertex v has a label $\Delta_v \geq 0$.
- $\Delta_v = 0$ if v is a leaf.
- If u is a child of v, then $\Delta_v \geq \Delta_u$
- $\Delta_{lca(u,v)}$ denotes the t-approximated distance between two leaves u and v

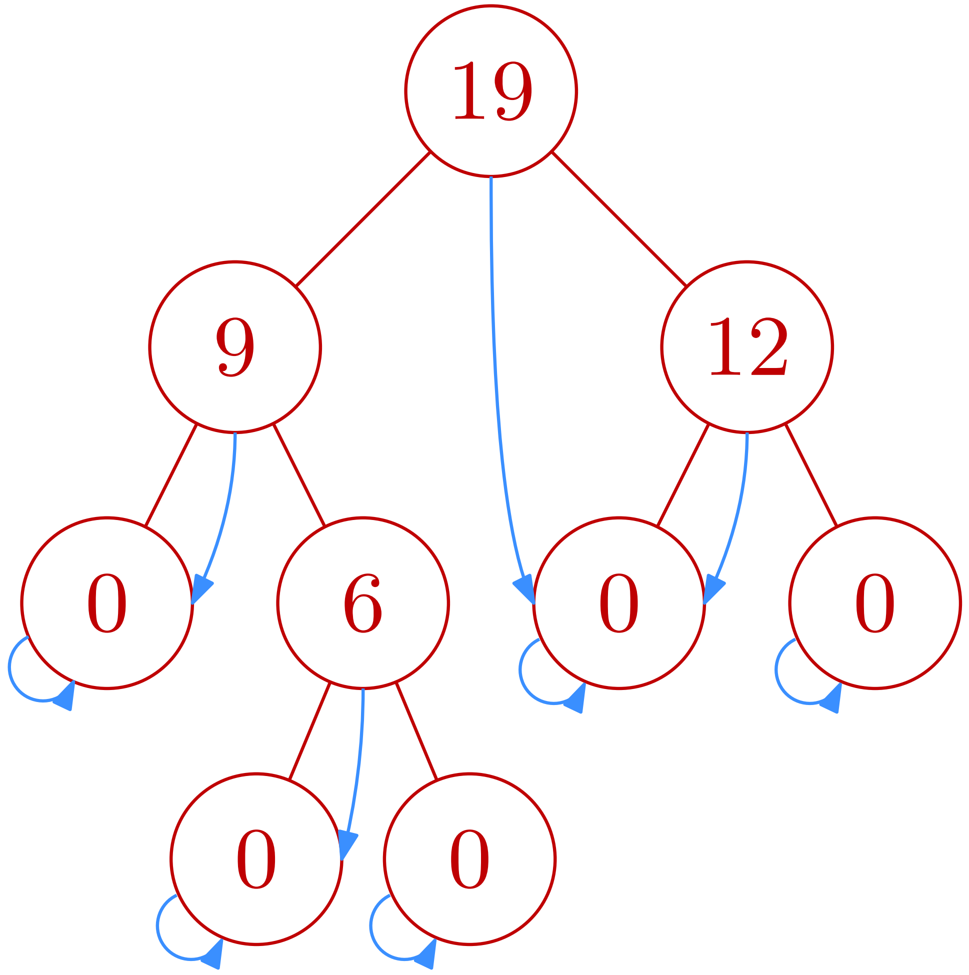
- Each vertex v has a representative leaf repr$_v$.
- repr$_u \in \{$repr$_v \mid$ v is a child of u$\}$



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

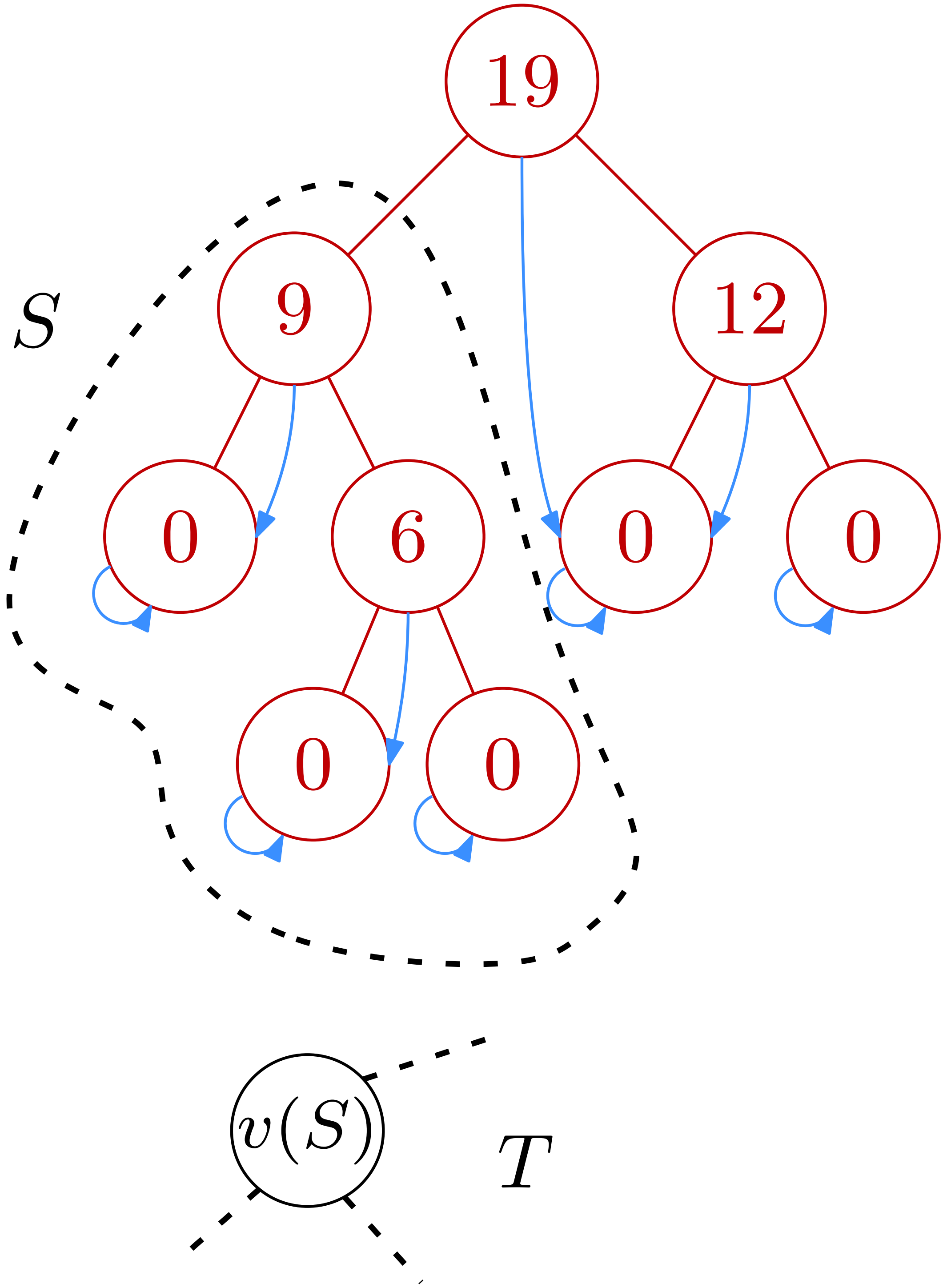


The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$



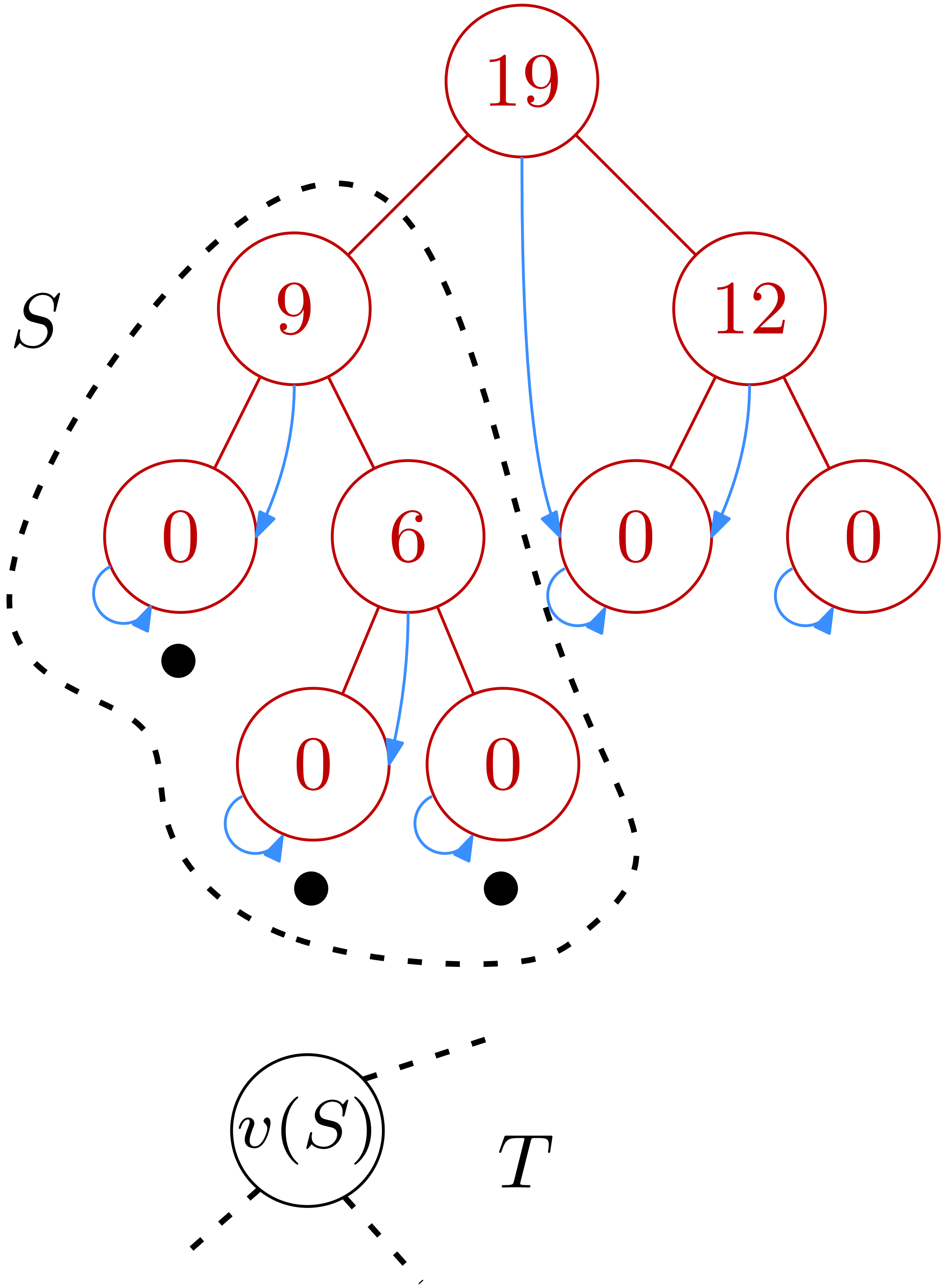
The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S



The ANN data structure

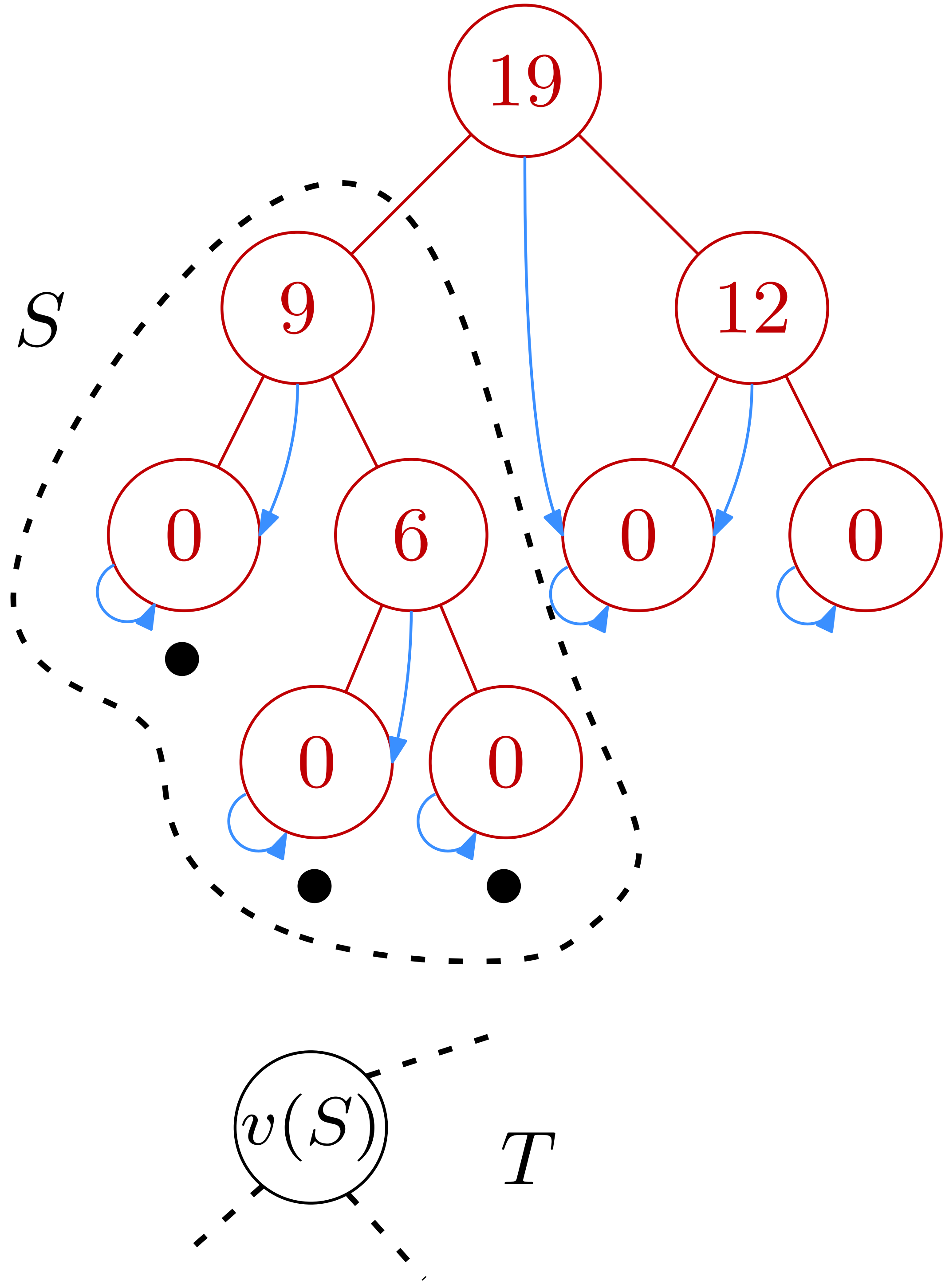
Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S



The ANN data structure

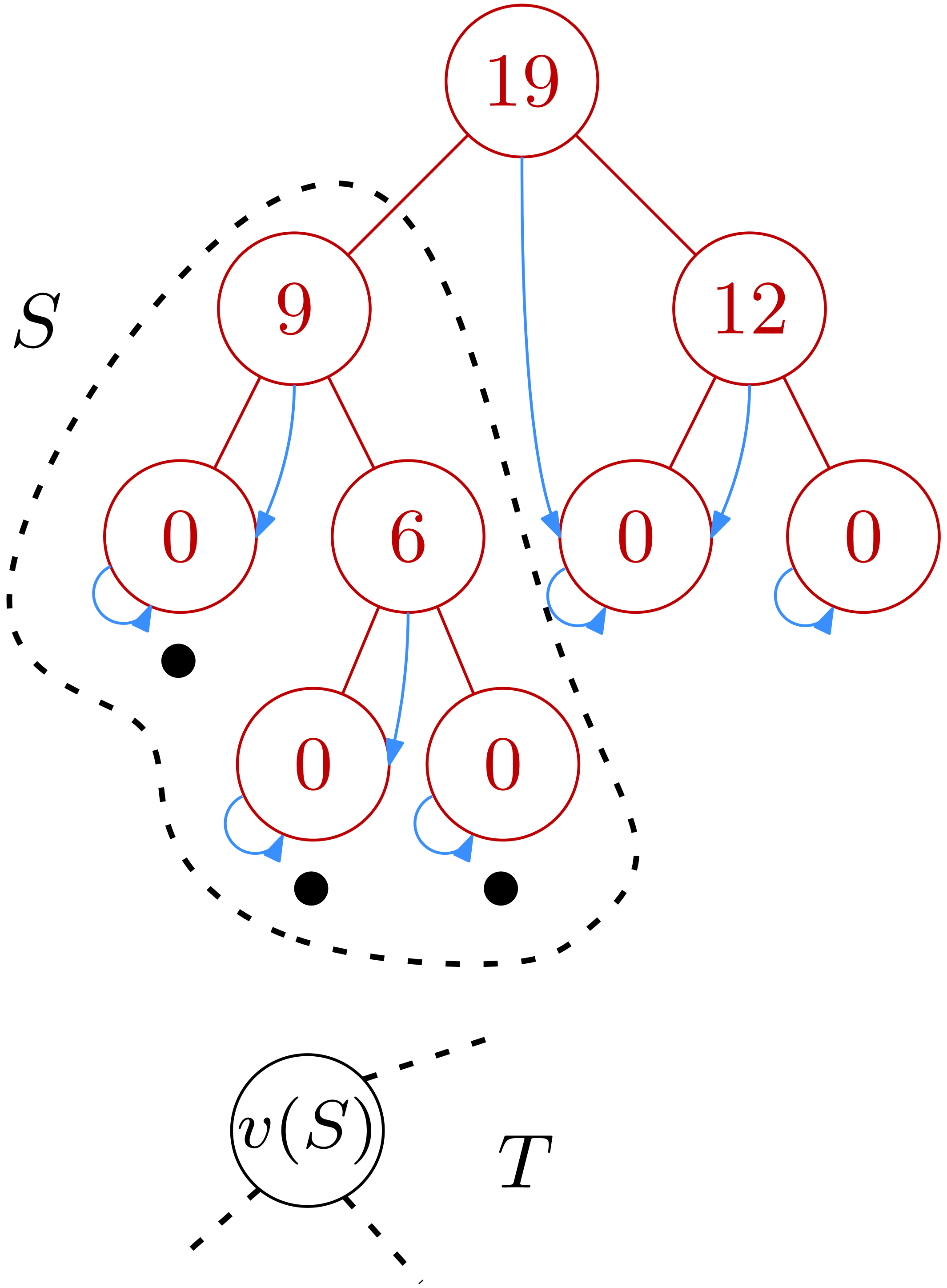
Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S
Removing a separator node breaks a (sub)tree T into connected components of size at most $|V(T)|/2$



The ANN data structure

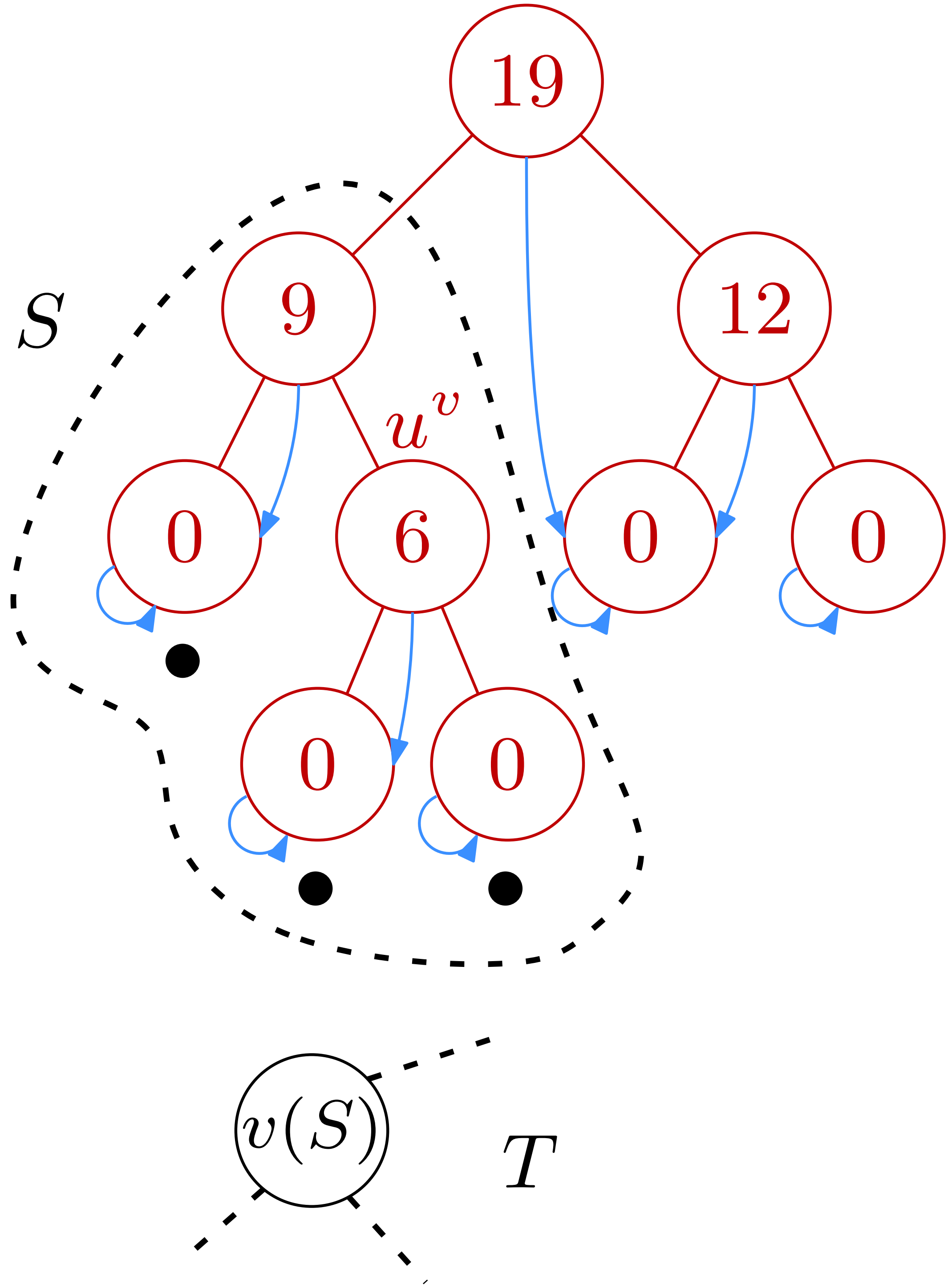
Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S
Removing a separator node breaks a (sub)tree T into
connected components of size at most $|V(T)|/2$



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

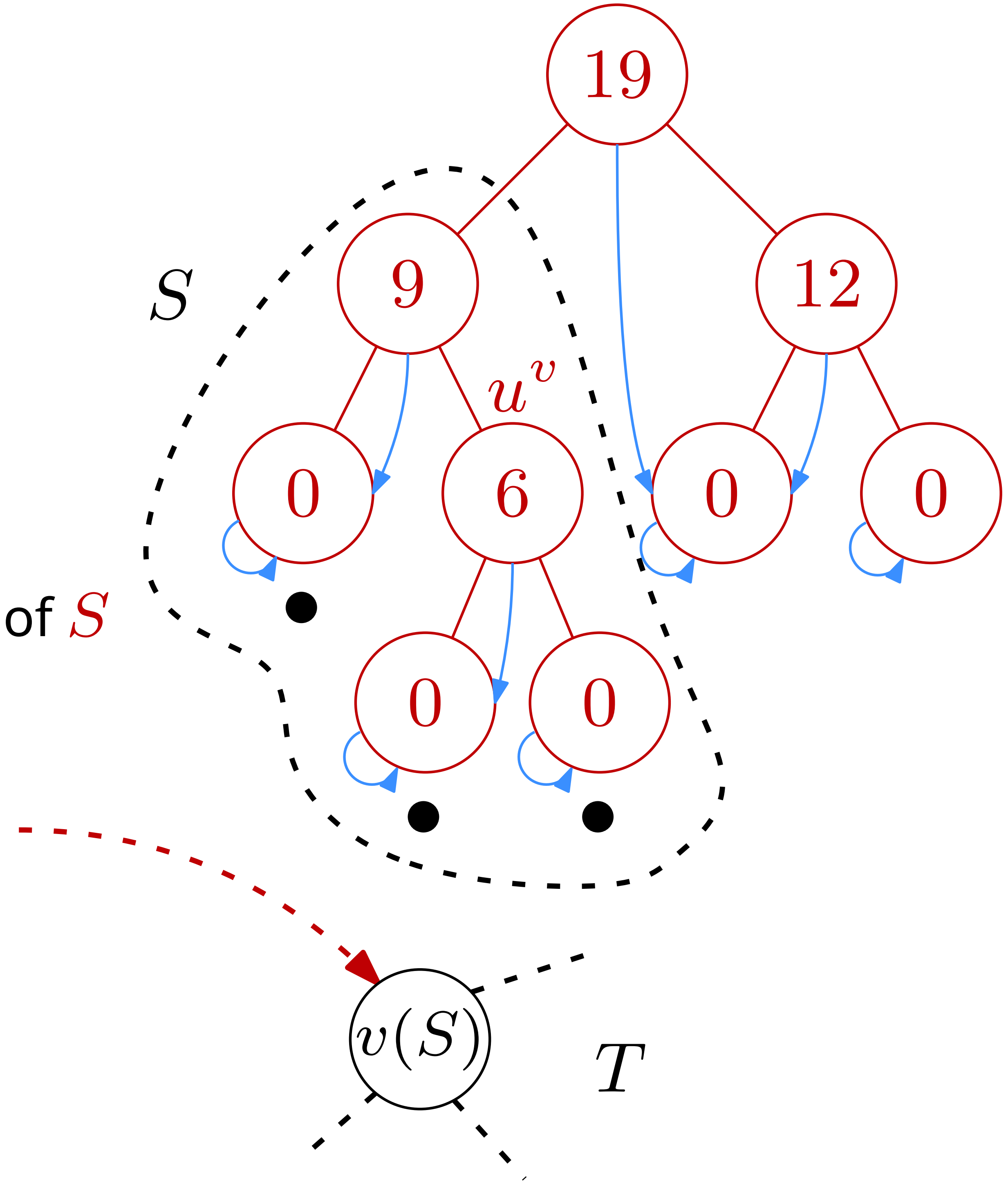
Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S

build and store $\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ in node v



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

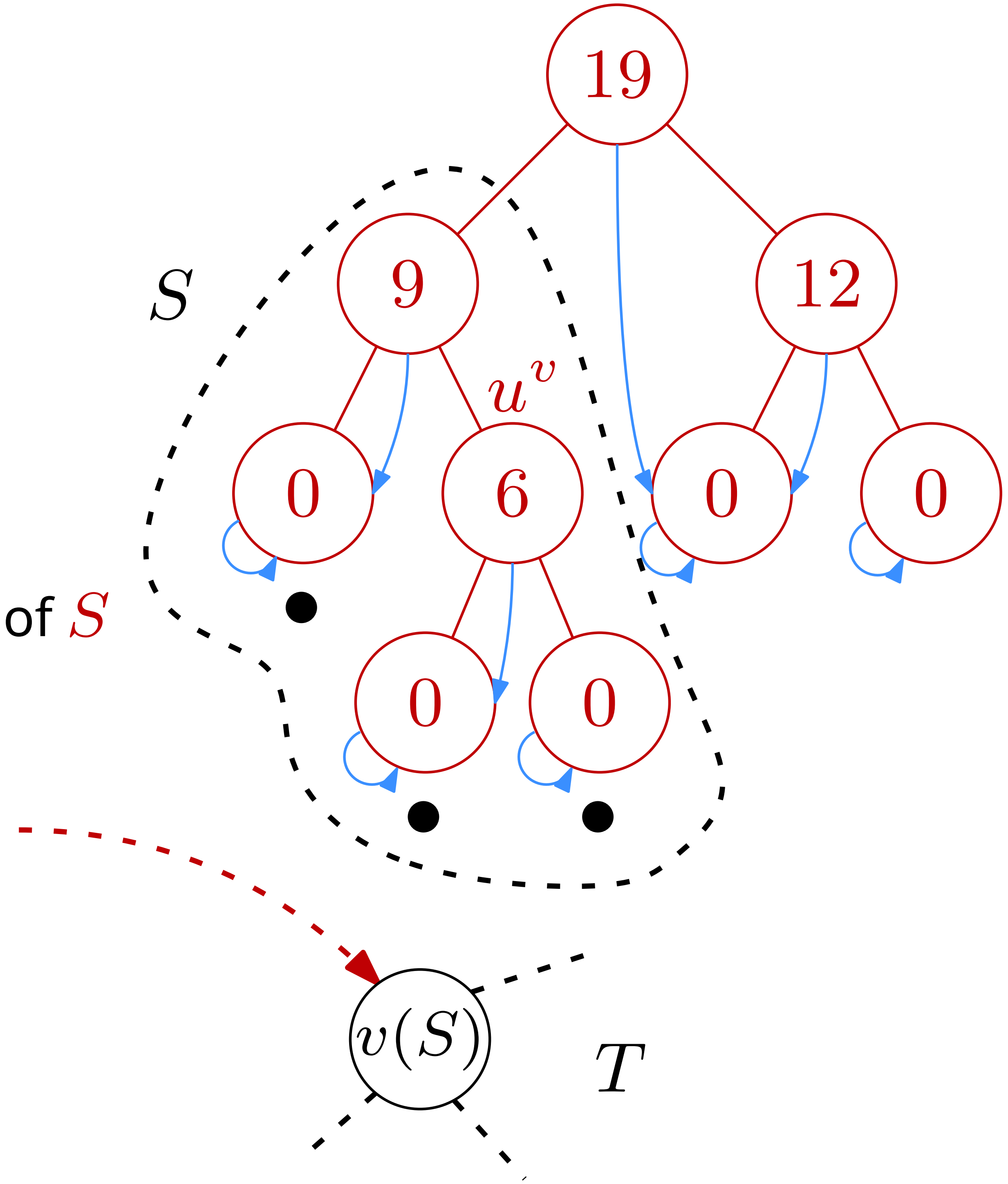
Given subtree S of H, create *searchtree* $T(S)$ rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S

build and store $\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ in node v

$$r_v = \frac{\Delta(u^v)}{4t} \text{ and } R_v = \mu\Delta(u^v)$$



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

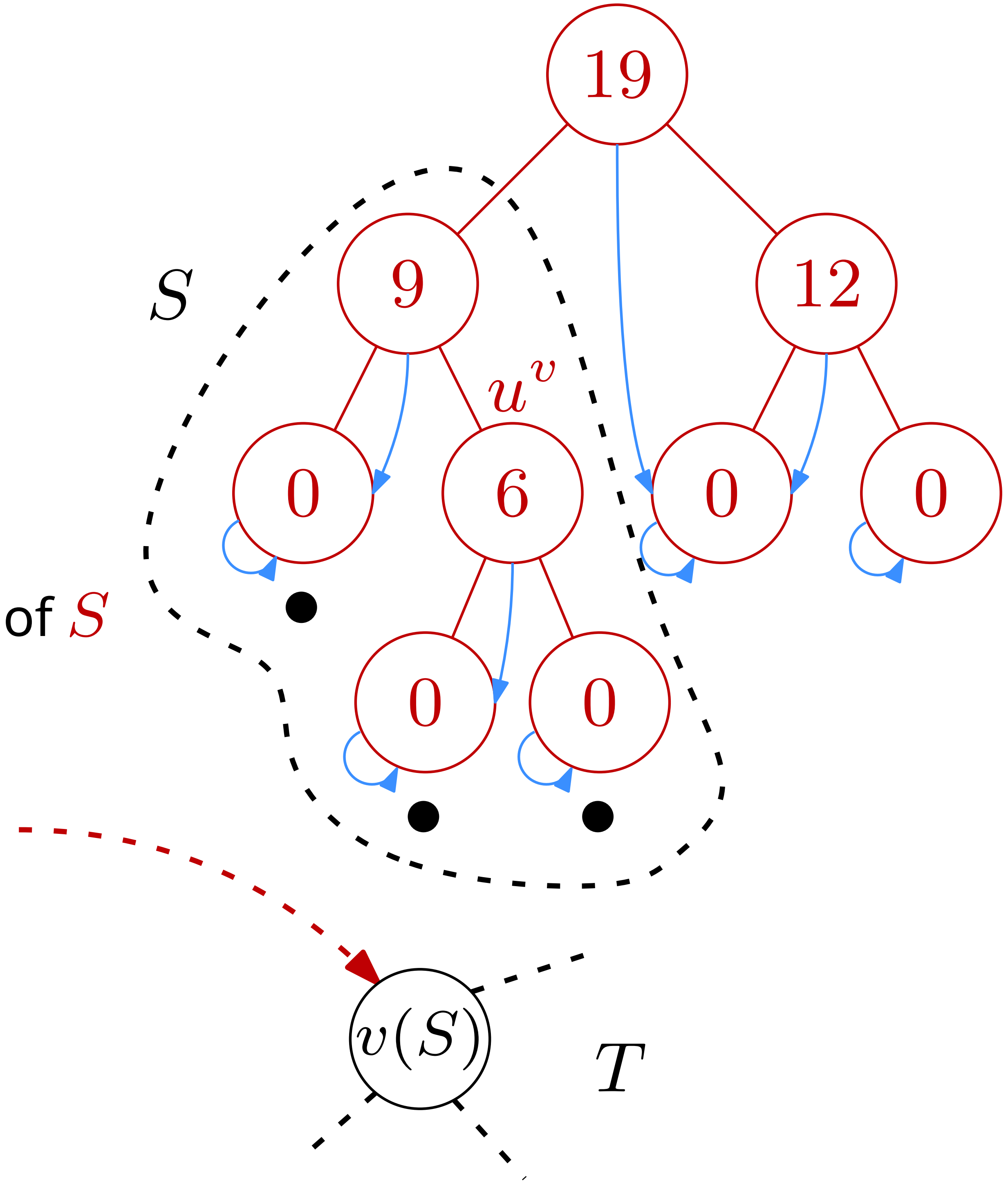
Let $P^v = P(S)$ be the set of **points/representatives** of S

Let $u^v \in V(S)$ be the separator of S

build and store $\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ in node v

$$r_v = \frac{\Delta(u^v)}{4t} \text{ and } R_v = \mu\Delta(u^v)$$

$$\mu = O(\varepsilon^{-1} \log n)$$



The ANN data structure

Given: set of points P and
a t-approximate (B)HST H on P

Recursively build search tree T (top-down):

Given subtree S of H, create *searchtree* $T(S)$
rooted at node $v = v(S) \in T$

Let $P^v = P(S)$ be the set of **points/representatives** of S

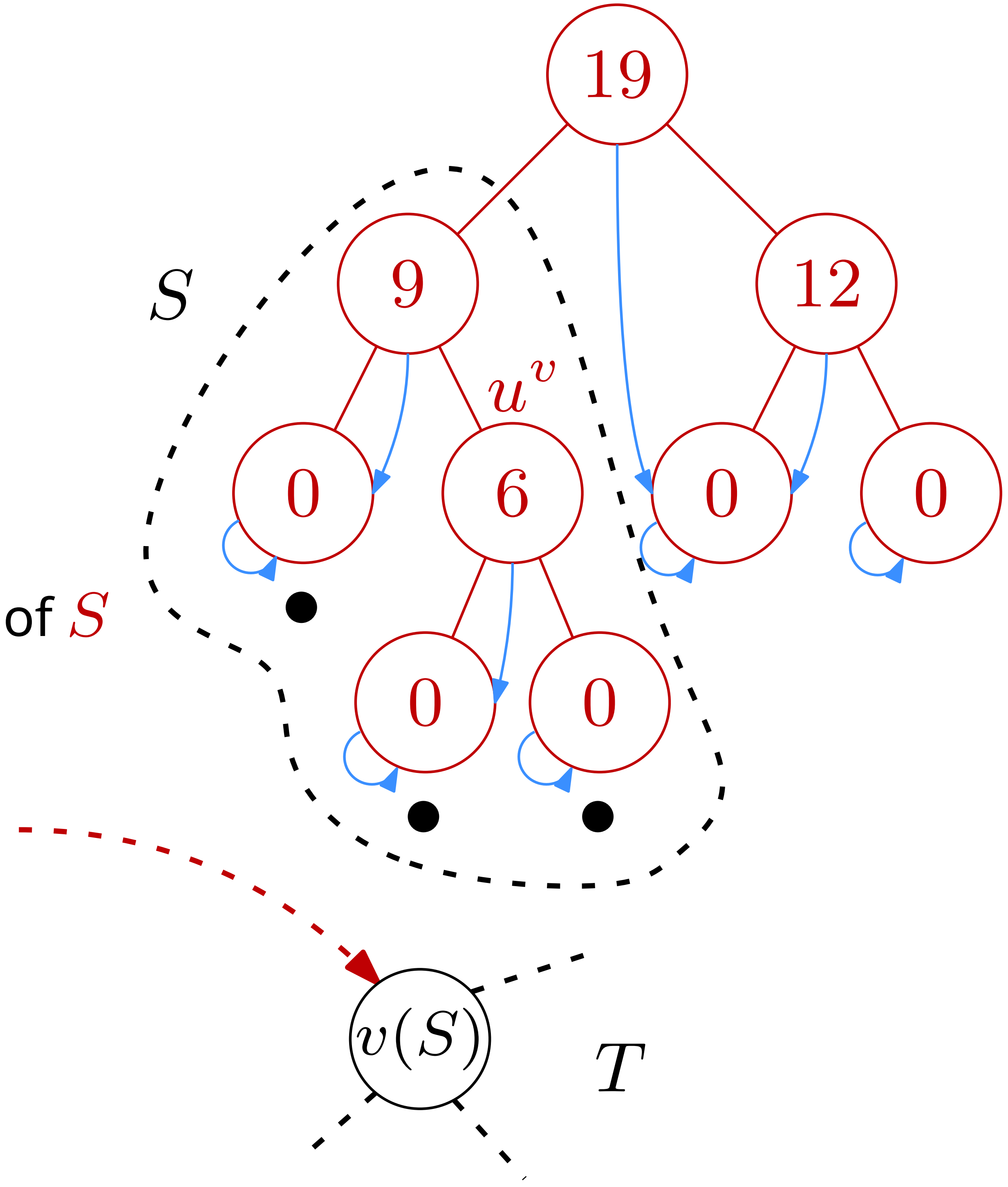
Let $u^v \in V(S)$ be the separator of S

build and store $\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ in node v

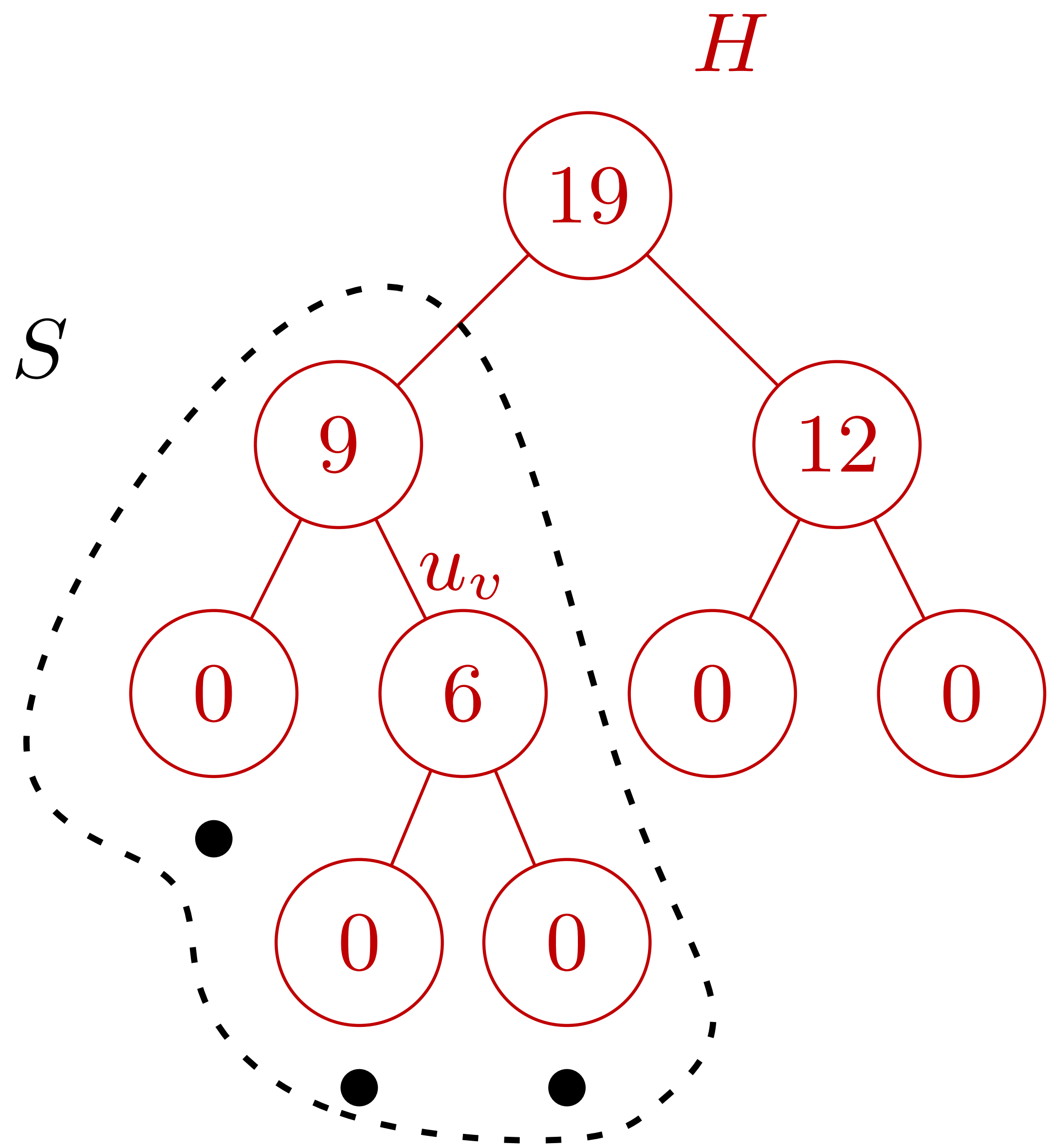
$$r_v = \frac{\Delta(u^v)}{4t} \text{ and } R_v = \mu\Delta(u^v)$$

$$\mu = O(\varepsilon^{-1} \log n)$$

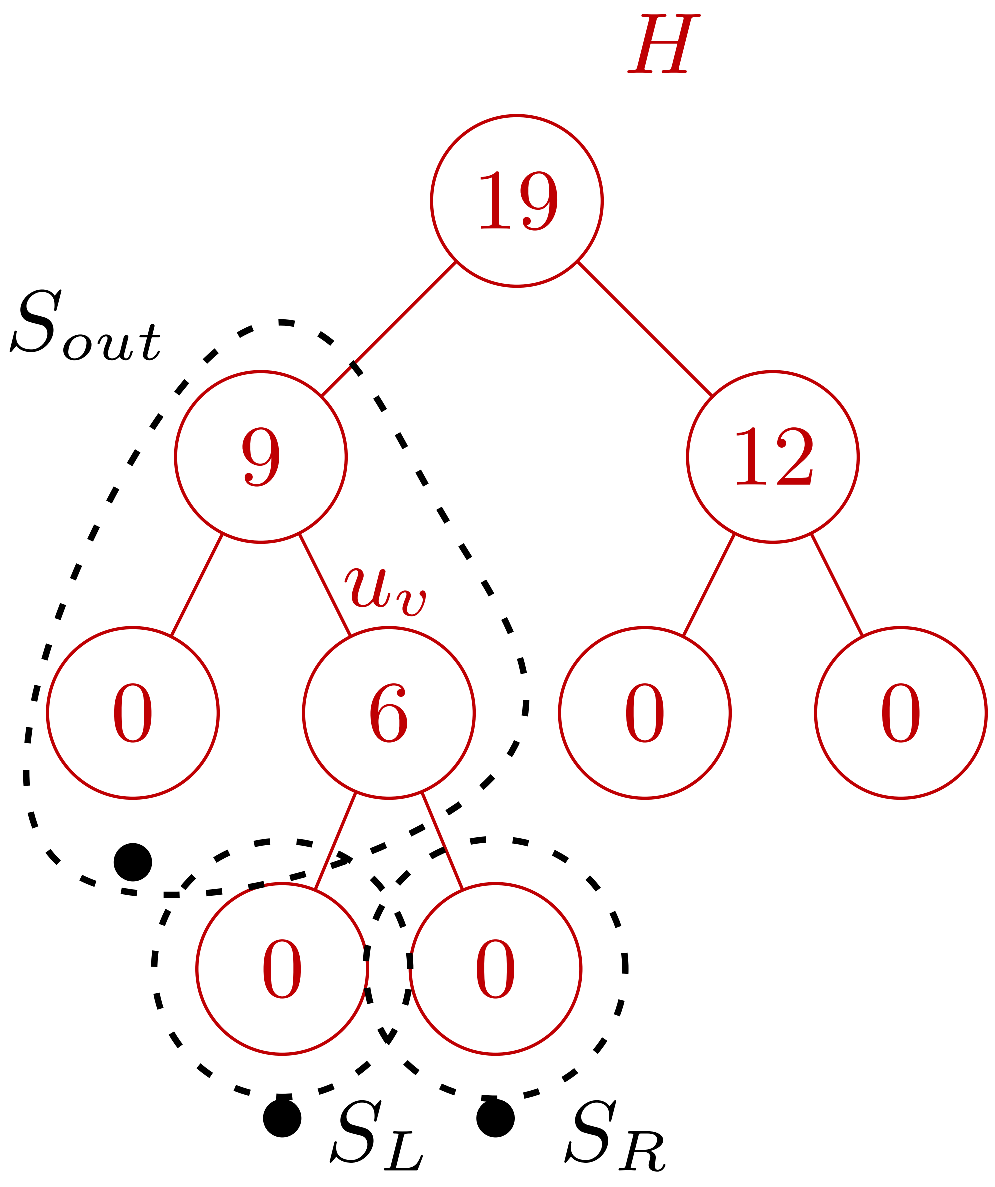
$\hat{\mathcal{I}}_v$ can be used to determine search path in T



The ANN data structure



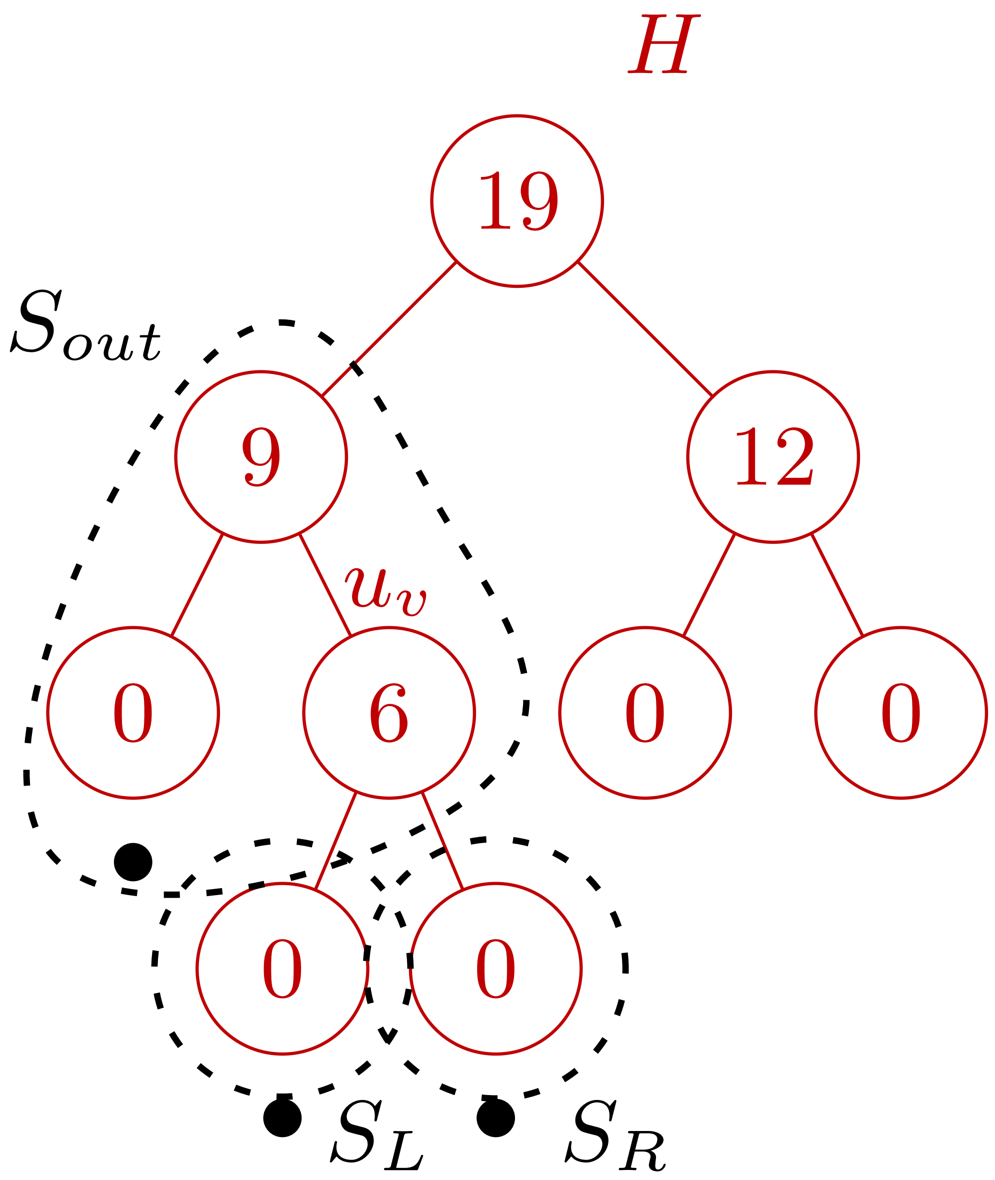
The ANN data structure



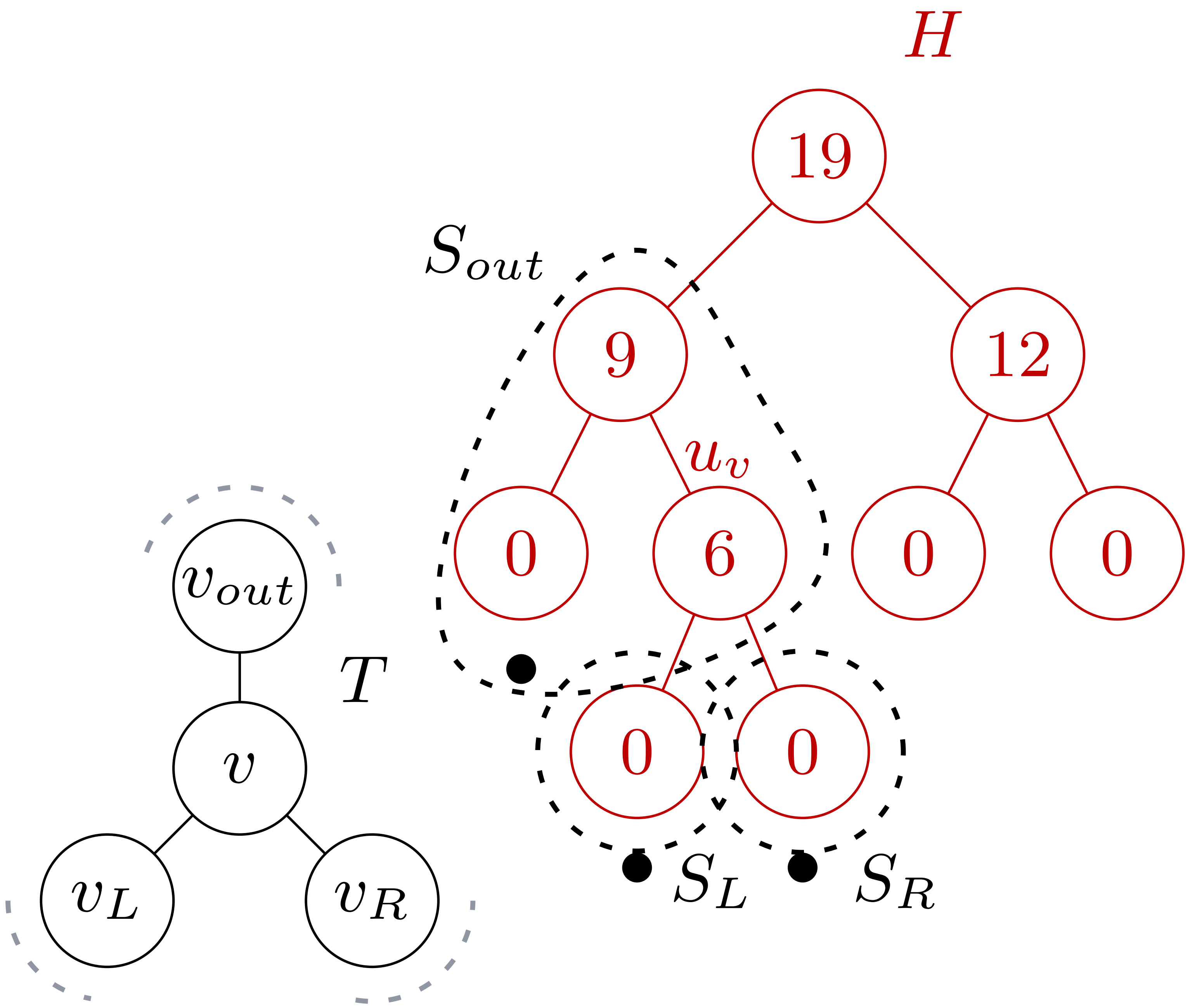
The ANN data structure

subtle:

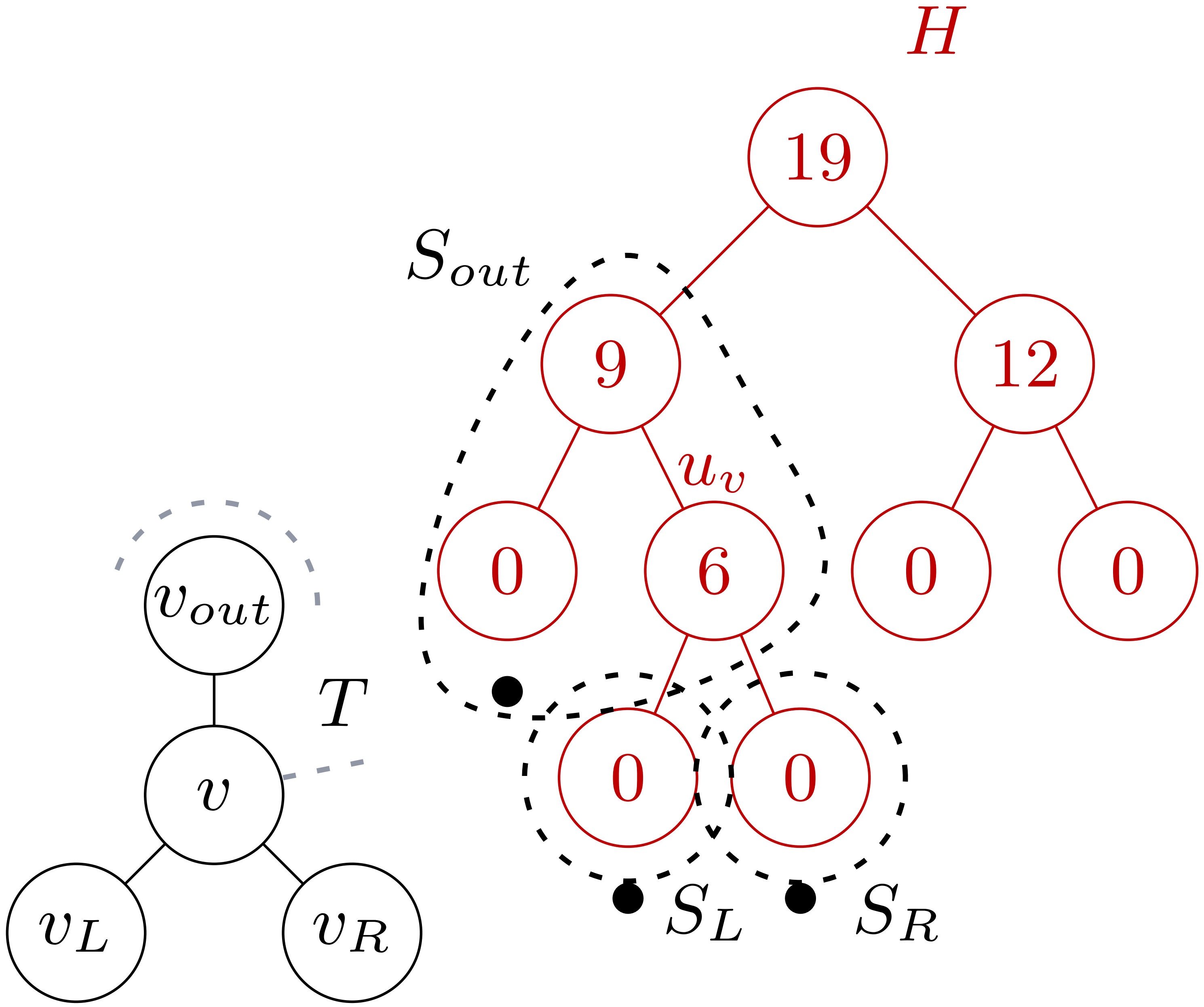
S_{out} contains rep_{u_v} but no other points of S_L and S_R.



The ANN data structure



The ANN data structure

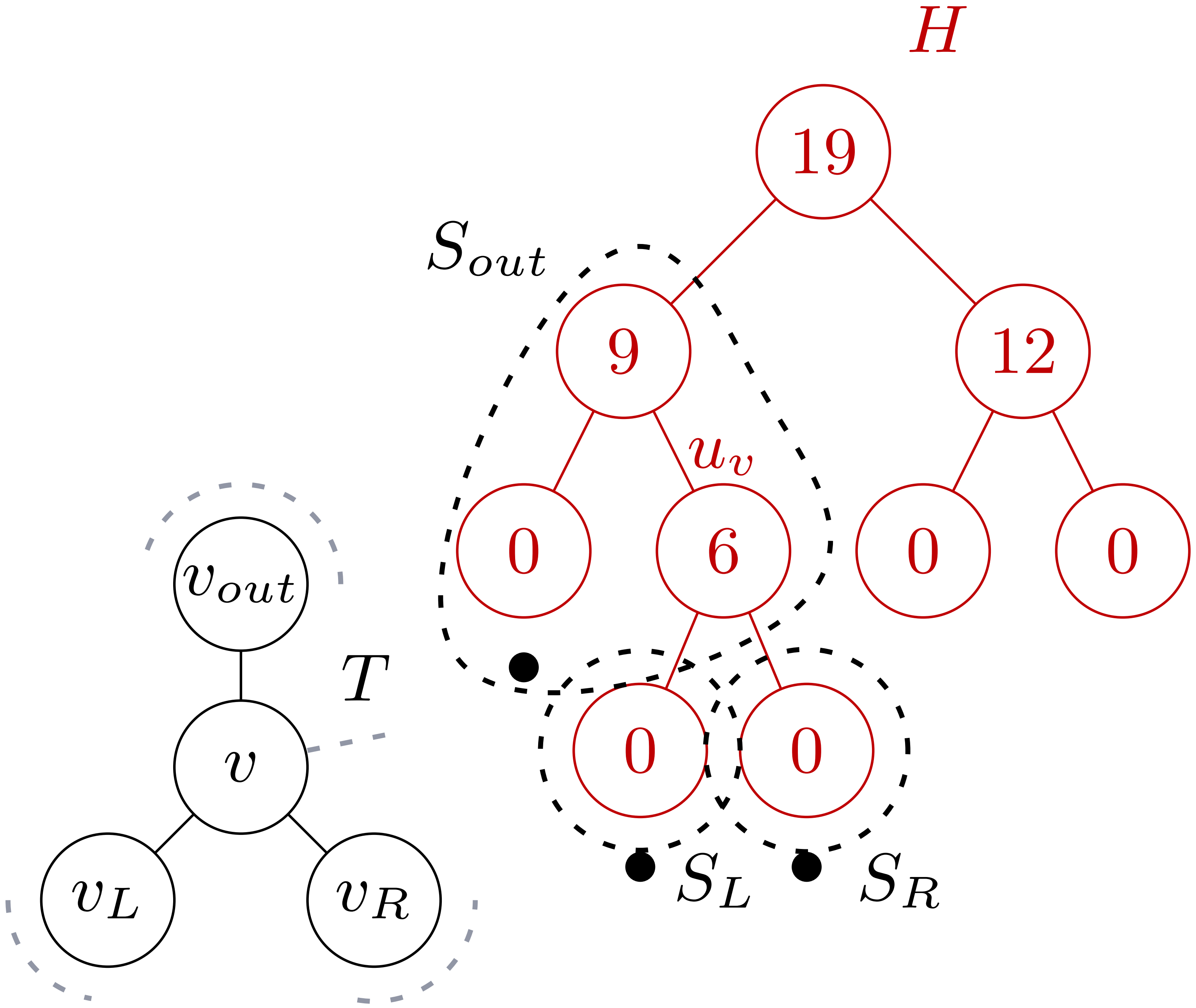


The ANN data structure

$\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ stored in node v

$r_v = \frac{\Delta(u^v)}{4t}$ and $R_v = \mu\Delta(u^v)$

A query into $\hat{\mathcal{I}}_v$ results in one of three cases:



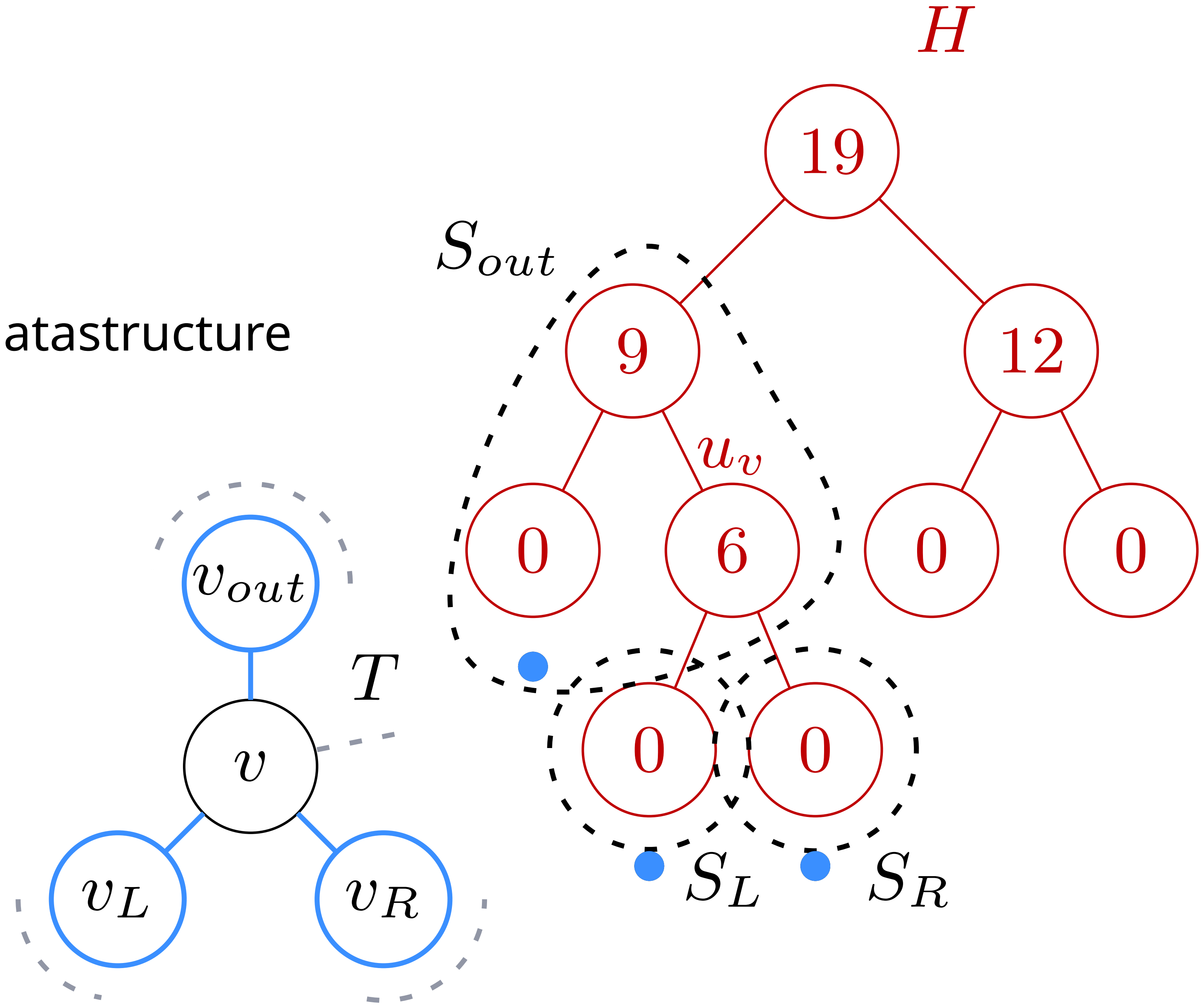
The ANN data structure

$\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ stored in node v

$$r_v = \frac{\Delta(u^v)}{4t} \text{ and } R_v = \mu\Delta(u^v)$$

A query into $\hat{\mathcal{I}}_v$ results in one of three cases:

- $d(q, P^v) \leq r_v$: Then $q \in \mathcal{U}(P^v, r_v)$ and the datastructure returns a point $p \in P^v$ with $d(q,p) \leq r_v$.
  Recurse into the subtree containing p



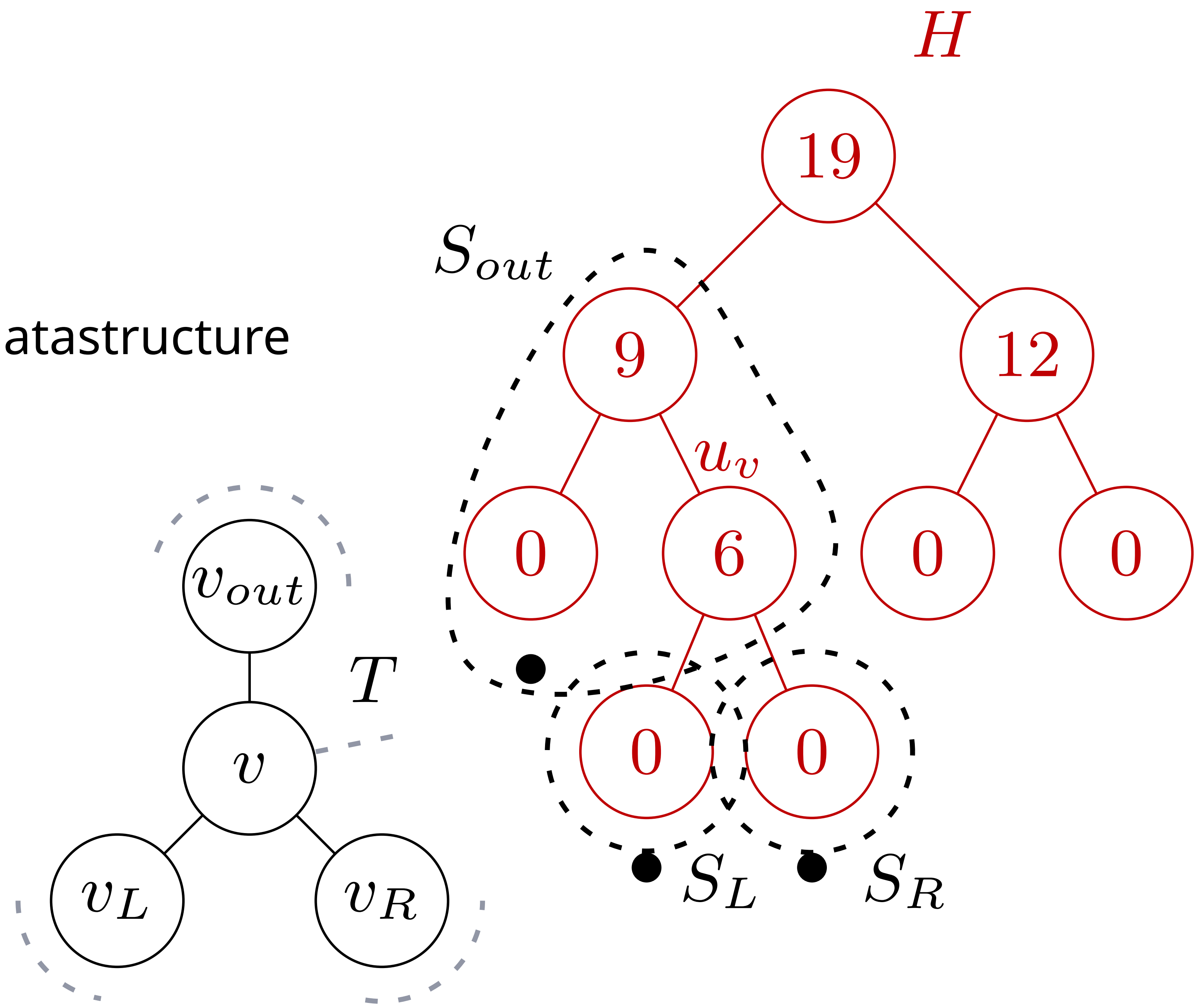
The ANN data structure

$\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ stored in node v

$r_v = \frac{\Delta(u^v)}{4t}$ and $R_v = \mu\Delta(u^v)$

A query into $\hat{\mathcal{I}}_v$ results in one of three cases:

- $d(q, P^v) \leq r_v$: Then $q \in \mathcal{U}(P^v, r_v)$ and the datastructure returns a point $p \in P^v$ with $d(q, p) \leq r_v$. Recurse into the subtree containing p
- $d(q, P^v) \in (r_v, R_v]$: Then the query finds a $(1 + \varepsilon/4)$-ANN point $s \in P^v$ and returns it as the answer to the query



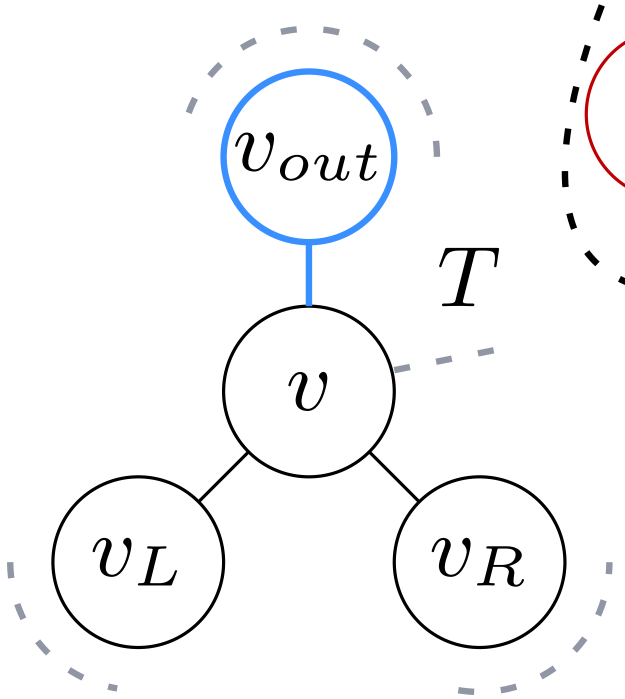
The ANN data structure

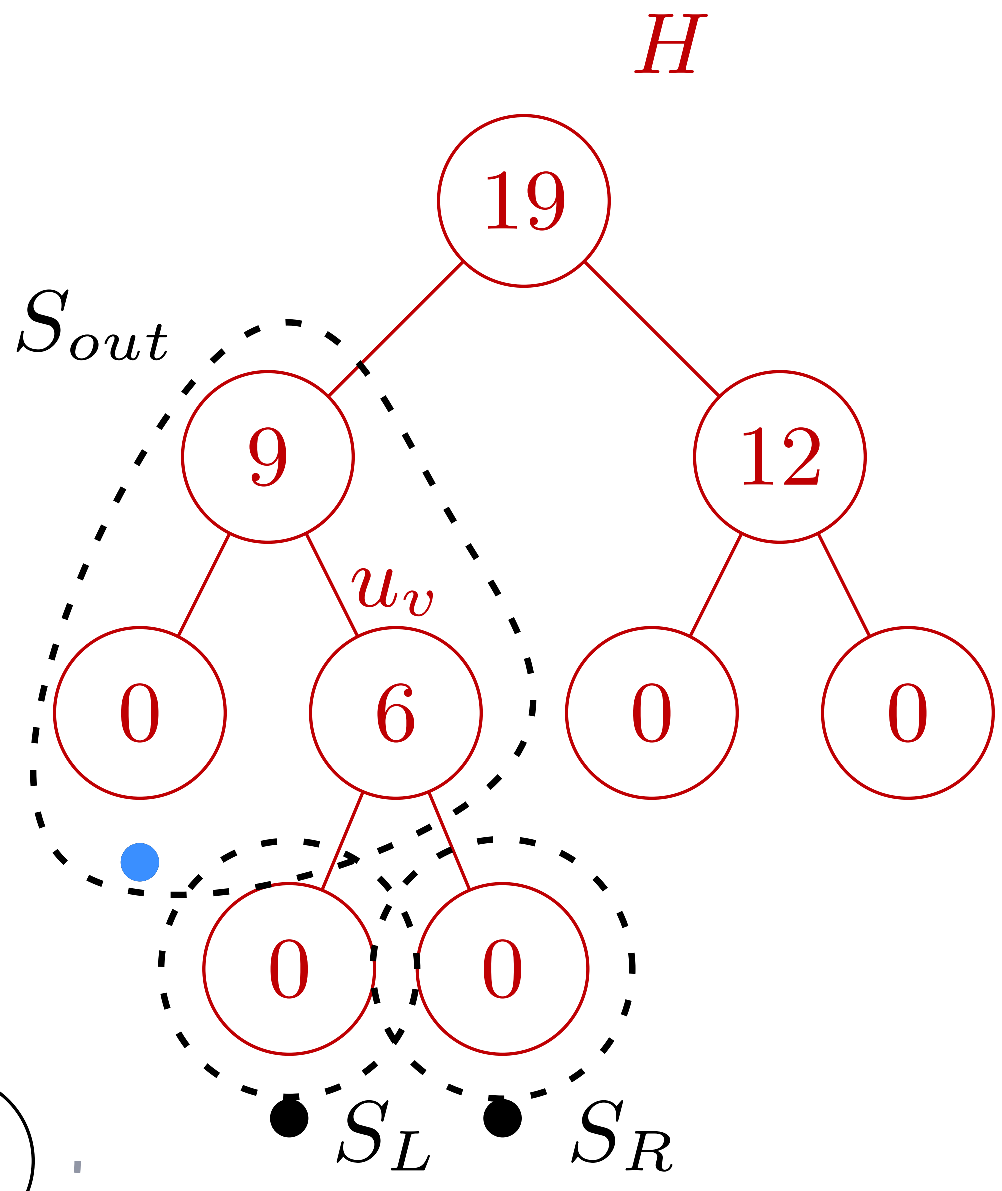
$\hat{\mathcal{I}}_v = \hat{\mathcal{I}}(P^v, r_v, R_v, \varepsilon/4)$ stored in node v

$$r_v = \frac{\Delta(u^v)}{4t} \text{ and } R_v = \mu\Delta(u^v)$$

A query into $\hat{\mathcal{I}}_v$ results in one of three cases:

- $d(q, P^v) \leq r_v$: Then $q \in \mathcal{U}(P^v, r_v)$ and the datastructure returns a point $p \in P^v$ with $d(q,p) \leq r_v$.
  Recurse into the subtree containing p
- $d(q, P^v) \in (r_v, R_v]$: Then the query finds a $(1+\varepsilon/4)$-ANN point $s \in P^v$ and returns it as the answer to the query
- $d(q, P^v) > R_v$:
  Then the search continues recursively in v_{out}





Correctness

Lemma: *The point returned by the data structure is a* $(1+\varepsilon)$*-ANN to the query point* q *in* P.

Proof (sketch, case 1):
$d(q, P^v) \leq r_v$; recurse into subtree with returned point p.

Correctness

Lemma: *The point returned by the data structure is a $(1+\varepsilon)$-ANN to the query point q in P.*

Proof (sketch, case 1):

$d(q, P^v) \leq r_v$; recurse into subtree with returned point p.

We need to show that the algorithm recurses into a subtree which contains a $(1+\varepsilon)$-ANN point. If the algorithm continues to v_L, we have that $d(P_L^v, P^v \setminus P_L^v) \geq \Delta(u^v)/t$. For $q_L = nn(q, P_L^v)$, by the triangle inequality we have that

$$d(q, P_v \setminus P_L^v) \geq d(q_L, P^v \setminus P_L^v) - d(q, q_L) \geq \frac{\Delta(u^v)}{t} - r_v > \frac{\Delta(u^v)}{2t} > r_v$$

$$r_v = \Delta(u^v)/4t$$

Space Complexity

Lemma: *For $t = n^{O(1)}$, the data structure is made out of $O(\frac{n}{\varepsilon}\log^2 n)$ balls.*

Proof (sketch):

Space Complexity

Lemma: *For* $t = n^{O(1)}$*, the data structure is made out of* $O(\frac{n}{\varepsilon}\log^2 n)$ *balls.*

Proof (sketch):

Let $U(n_v)$ be the number of balls used in $\hat{\mathcal{I}}_v$, we have $\mu = O(\varepsilon^{-1}\log n)$ and $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ is made out of $O(\frac{n}{\varepsilon}\log(b/a))$ balls (previous slide).

Space Complexity

Lemma: *For $t = n^{O(1)}$, the data structure is made out of $O(\frac{n}{\varepsilon} \log^2 n)$ balls.*

Proof (sketch):

Let $U(n_v)$ be the number of balls used in $\hat{\mathcal{I}}_v$, we have $\mu = O(\varepsilon^{-1} \log n)$ and $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ is made out of $O(\frac{n}{\varepsilon} \log(b/a))$ balls (previous slide).

we have:

$$U(n_v) = O(\frac{n_v}{\varepsilon} \log \frac{R_v}{r_v}) = O(\frac{n_v}{\varepsilon} \log(\frac{t \log n}{\varepsilon}))$$

Space Complexity

Lemma: *For $t = n^{O(1)}$, the data structure is made out of $O(\frac{n}{\varepsilon}\log^2 n)$ balls.*

Proof (sketch):

Let $U(n_v)$ be the number of balls used in $\hat{\mathcal{I}}_v$, we have $\mu = O(\varepsilon^{-1}\log n)$ and $\hat{\mathcal{I}}(P, a, b, \varepsilon)$ is made out of $O(\frac{n}{\varepsilon}\log(b/a))$ balls (previous slide).

we have:

$$U(n_v) = O(\frac{n_v}{\varepsilon}\log\frac{R_v}{r_v}) = O(\frac{n_v}{\varepsilon}\log(\frac{t\log n}{\varepsilon}))$$

count each occurrence of point as repr. (overall $2n$)

For the total number of balls, we get the recurrence
$B(n) = U(n) + B(n_L) + B(n_R) + B(n_{out})$
(we have $n_L, n_R, n_{out} \leq n/2 + 1$ and $B(n_L) + B(n_R) + B(n_{out}) = n$).
This results in $B(2n) = O(\varepsilon^{-1} n \log n \log(\varepsilon^{-1} t \log n)) = O((n/\varepsilon)\log^2 n)$

Number of Queries

Lemma: *For* $t = n^{O(1)}$*, the ANN-query algorithm performs* $O(\log(n/\varepsilon))$ *near neighbor queries*

Proof (sketch):

Number of Queries

Lemma: *For* $t = n^{O(1)}$*, the ANN-query algorithm performs* $O(\log(n/\varepsilon))$ *near neighbor queries*

Proof (sketch):

For every internal node v on the search path π in T corresponds to a situation where $d(q, P) \leq r_v$ or $d(q, P) > R_v$, which can be decided by two nearest neighbor queries.

In the final node u in π, the search algorithm resolves the query using $O\left(\log \frac{\log R_v / r_v}{\varepsilon}\right)$ $= O(\log(1/\varepsilon) + \log\log n)$ near neighbor queries.

Since the depth of the tree is $O(\log n)$, the total number of queries becomes $O(\log n + \log(1/\varepsilon) + \log\log n) = O(\log n/\varepsilon)$.

Summary

Approximating a metric space by a hierarchical well-separated tree (HST)

hierachical well-separated trees

simple $(n-1)$-approximation

fast $n^{O(1)}$-approximation in $\mathbb{R}^d$

ANN via point location among balls

simple construction

handling a range of radii

ANN data structure based on HST

. . . **and next time**

point location among approximate balls

approximate Voronoi diagrams