Nearest Neighbour Searching in Metric Spaces

Kenneth Clarkson (1999, 2006)

Nearest Neighbour Search

Problem **NN**

- Given:
  - Set U
  - Distance measure D
  - Set of sites $S \subset U$
  - Query point $q \in U$
- Find:
  - Point $p \in S$ such that $D(p, q)$ is minimum

Outline

- Applications and variations
- Metric Spaces
  - Basic inequalities
- Basic algorithms
  - Orchard, annulus, AESA, metric trees
- Dimensions
  - Coverings, packings, ε-nets
  - Box, Hausdorff, packing, pointwise, doubling dimensions
  - Estimating dimensions using NN
- NN using dimension bounds
  - Divide and conquer
    - Exchangeable queries
  - $M(S, Q)$ and auxiliary query points

Applications

- "Post-office problem"
  - Given a location on a map, find the nearest post-office/train station/restaurant...
- Best-match file searching (key search)
- Similarity search (databases)
- Vector quantization (information theory)
  - Find codeword that best approximates a message unit
- Classification/clustering (pattern recognition)
  - e.g. **k-means clustering** requires a nearest neighbour query for each point at each step

Variations

- k-nearest neighbours
  - Find k sites closest to query point q
- Distance range searching
  - Given query point q, distance r, find all sites $p \in S$ s.t. $D(q, p) \leq r$
- All (k) nearest neighbours
  - For each site s, find its (k) nearest neighbour(s)
- Closest pair
  - Find sites s and s' s.t. D(s, s') is minimized over S

Variations

- Reverse queries
  - Return each site with q as its nearest neighbour in $S \cup \{q\}$ (excluding the site itself)
- Approximate queries
  - (δ)-nearest neighbour
    - Any point whose distance to q is within a δ factor of the nearest neighbour distance
  - Interesting because approximate algorithms usually achieve **better running times** than exact versions
- Bichromatic queries
  - Return closest red-blue pair

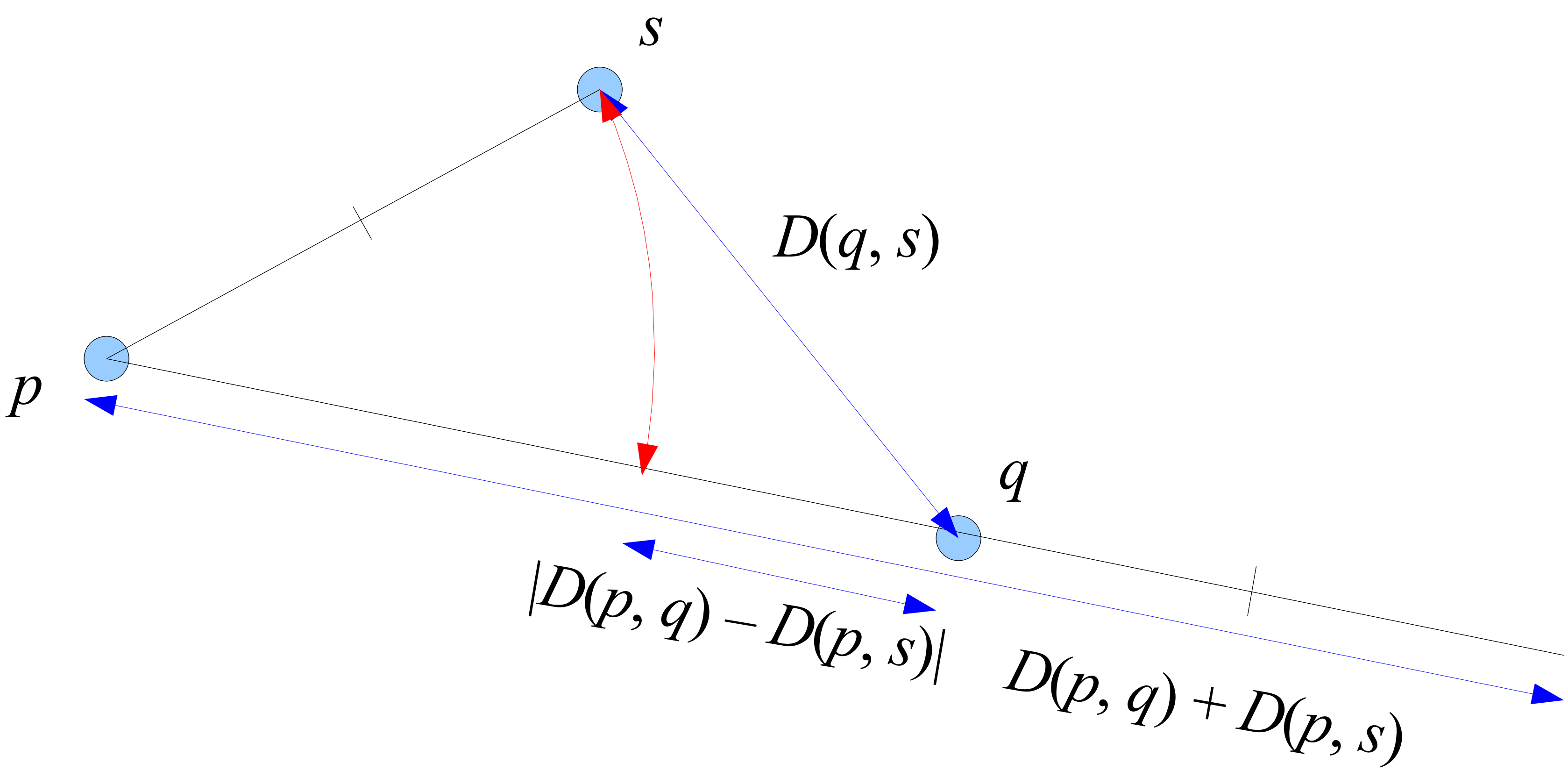
Metric Spaces

- Metric space $Z := (U, D)$
  - Set U
  - Distance measure D
- D satisfies
  1. Nonnegativity: $D(x, y) \geq 0$
  2. Small self-distance: $D(x, x) = 0$
  3. Isolation: $x \neq y \Rightarrow D(x, y) > 0$
  4. Symmetry: $D(x, y) = D(y, x)$
  5. Triangle inequality: $D(x, z) \leq D(x, y) + D(y, z)$
- Absence of any **one** of 3-5 can be “repaired”.

Triangle Inequality Bounds

For q, s, $p \in U$, any value r, and any $P \subset U$

1. $|D(p, q) - D(p, s)| \leq D(q, s) \leq D(p, q) + D(p, s)$



Triangle Inequality Bounds

2. $D(q, s) \geq D_P(q, s) := \max_{p \in P} |D(p, q) - D(p, s)|$

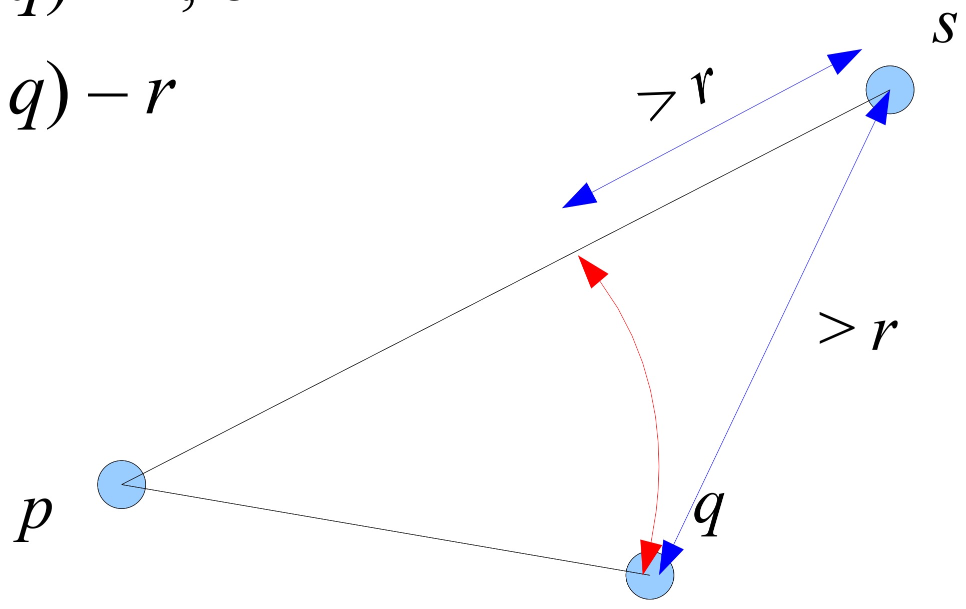
3. If

   $D(p, s) > D(p, q) + r$, or

   $D(p, s) < D(p, q) - r$

   Then

   $D(q, s) > r$



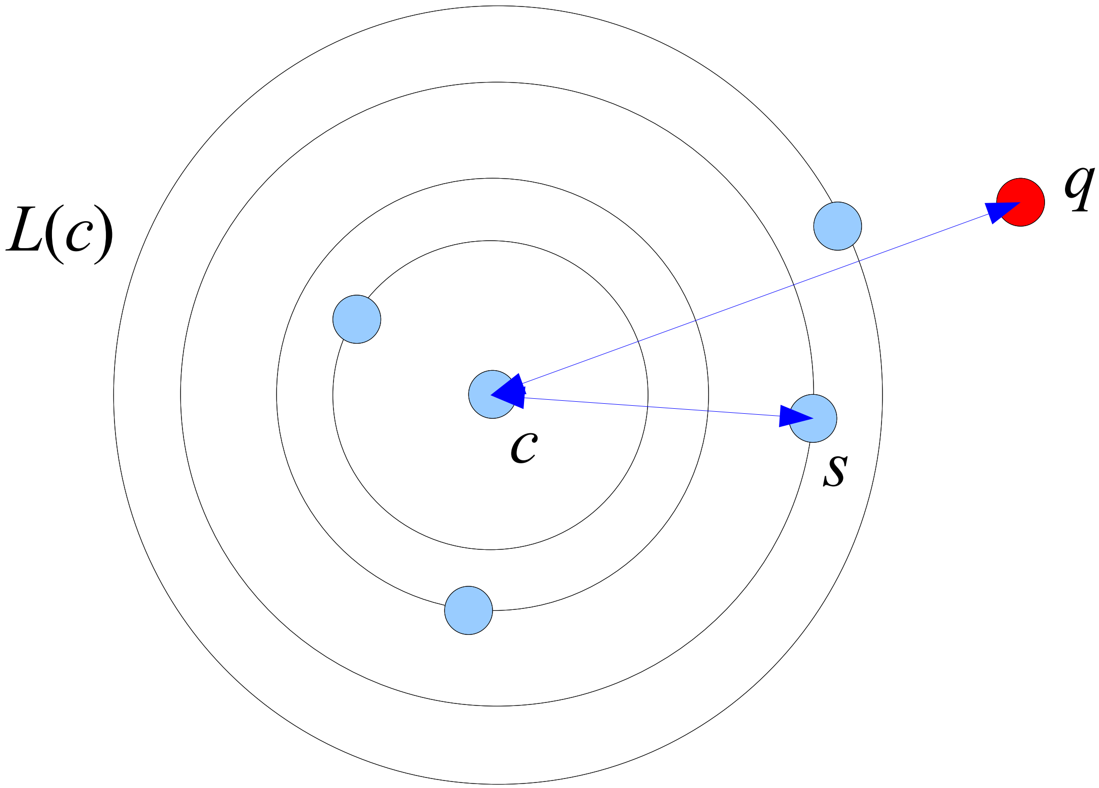
4. If $D(p, s) \geq 2D(p, q)$, then $D(q, s) \geq D(q, p)$

Triangle Inequality Bounds

- Utility: Give useful **stopping criteria** for NN searches
- Used by:
  - Orchard's Algorithm
  - Annulus Method
  - AESA
  - Metric Trees

Orchard's Algorithm

- For each site p, create a list of sites $L(p)$ in increasing order of distance to p
- Pick an initial candidate site c
- Walk along $L(c)$ until a site s nearer to q is found

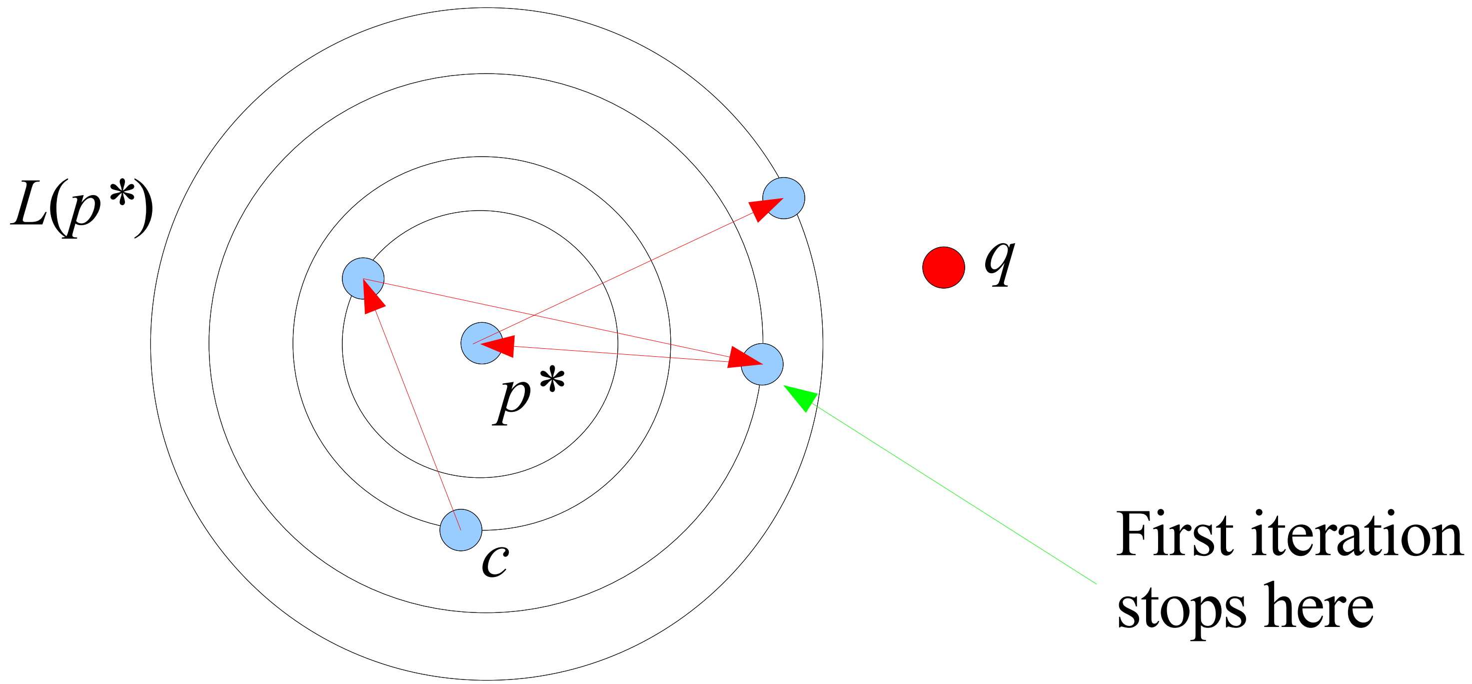


Orchard's Algorithm

- Make s the new candidate: $c := s$, and repeat
- Stopping criterion:
    - $L(c)$ is completely traversed for some c, or
    - $D(c, s) > 2D(c, q)$ for some s in $L(c)$
      $\Rightarrow D(s', q) > D(c, q)$ for all subsequent s' in $L(c)$ by Triangle Inequality Bound (4)
  - In either case, c is the nearest neighbour of q
- Performance:
  - $\Omega(n^2)$ preprocessing and storage – BAD!
- Refinement: Mark each site after it has been rejected
  - Ensures distance computations are reduced

Annulus Method

- Similar to Orchard's Algorithm, but uses linear storage
- Maintain just one list of sites $L(p^*)$ in order of increasing distance from a single (random) site p^*
- Pick an initial candidate site c
- Alternately move away from and towards p^*



Annulus Method

- If a site s closer to q than c is found, make s the new candidate: $c := s$, and repeat
- Stopping criterion:
  - A site s on the "lower" side has

    $$D(p^*, s) < D(p^*, q) - D(c, q),$$

    in which case we can ignore all lower sites
  - A site s on the "higher" side has

    $$D(p^*, s) > D(p^*, q) + D(c, q),$$

    in which case we can ignore all higher sites

  (Triangle Inequality Bound (3))

  – Stop when L(p*) is completely traversed – the final candidate is the nearest neighbour

AESA

- "Approximating and Eliminating Search Algorithm"
- Precomputes and stores distances $D(x, y)$ for all $x, y \in S$
- Uses lower bound $D_P(x, q)$
  - Recall: $D_P(x, q) := \max_{p \in P} |D(p, x) - D(p, q)| \leq D(x, q)$
- Every site x is in one of three states:
  - ***Known***: $D(x, q)$ has been computed
    - The known sites form a set P
  - ***Unknown***: Only a lower bound $D_P(x, q)$ is available
  - ***Rejected***: $D_P(x, q)$ is larger than distance of closest *Known* site

AESA

- Initial state: for each site x
  - x is *Unknown*
  - $D_P(x, q) = \infty$
- Repeat until all sites are *Known* or *Rejected*
  - Pick *Unknown* site with smallest $D_P(x, q)$ (break ties at random)
  - Compute $D(x, q)$, so x becomes *Known*
  - Update smallest distance r known to q
  - Set $P := P \cup \{x\}$, and for all *Unknown* x', update $D_P(x', q)$; make x' *Rejected* if $D_P(x, q) > r$
    - The update is easy since
      $$\mathrm{D}_{P \cup \{x\}}(x', q) = \max\{D_P(x', q), |D(x, q) - D(x, x')|\}$$

AESA

- Performance:
  - Average constant number of distance computations
  - $\Omega(n^2)$ preprocessing and storage

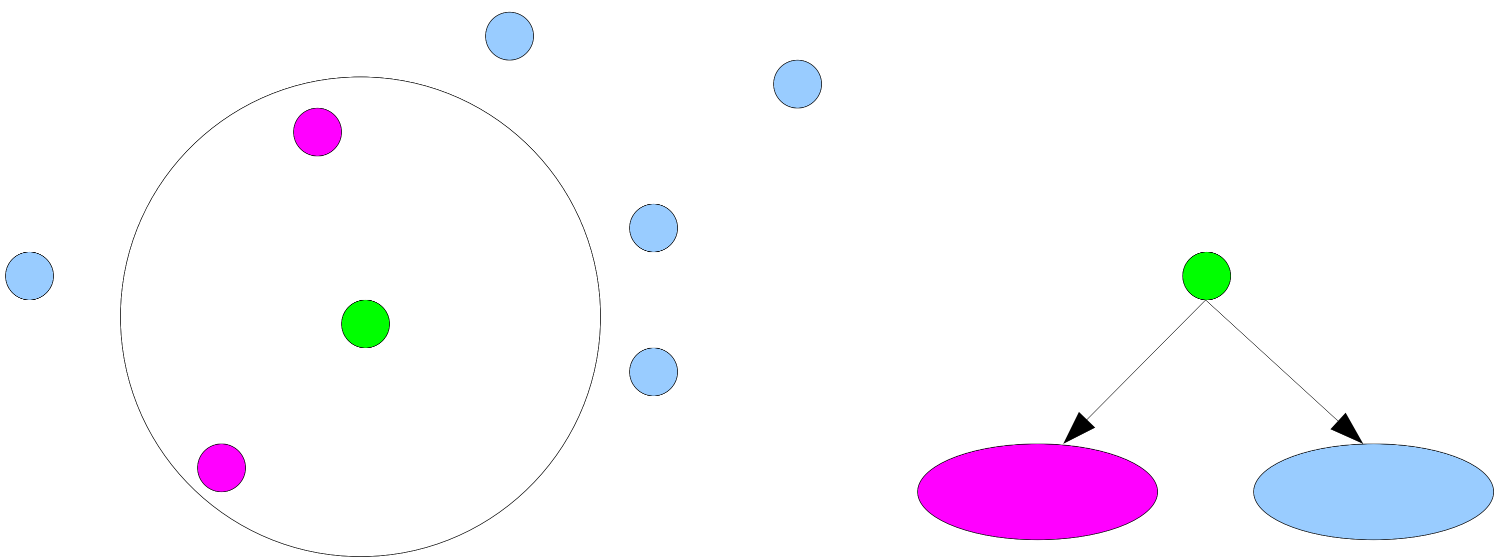
- Can we do better?
  - Yes! Linear AESA uses a constant-sized pivot set
  - [Mico, Oncina, Vidal '94]

Linear AESA

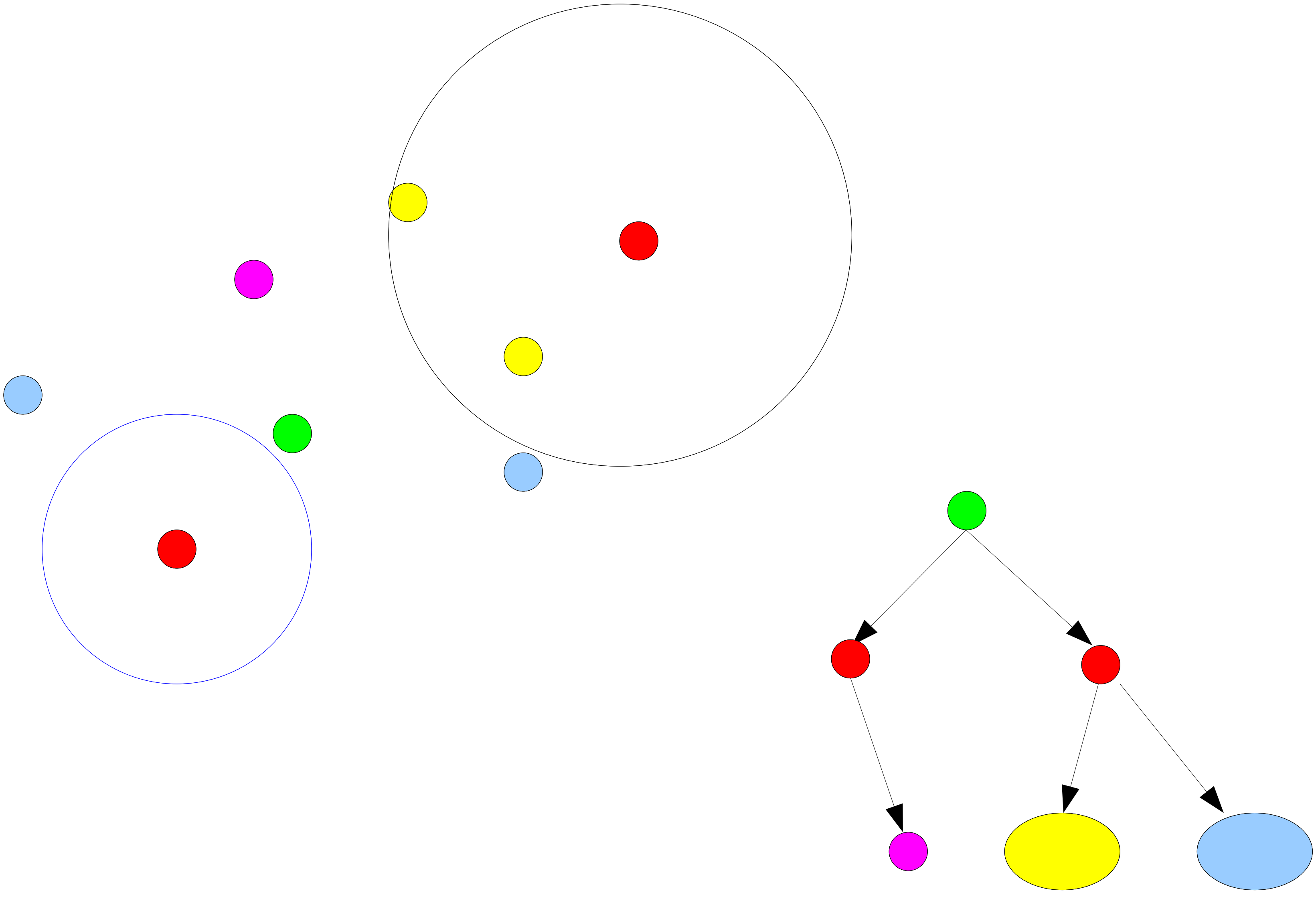
- Improvement: Use a subset V of the states, called “pivots”
- Let P only consist of pivots, and update it only when x is a pivot itself
  - Hence, only store distances to pivots
- For a constant sized pivot set, the preprocessing and storage requirements are linear
- Works best when pivots are well-separated
  - A greedy procedure based on “accumulated distances” is described in [Mico, Oncina, Vidal '94]
  - Similar to ε-nets?

Metric Trees

- Choose a seed site, construct a ball B around it, divide sites into two sets $S \cap B$ and $S \setminus B$ ("inside" and "outside") and recurse
- For suitably chosen balls and centres, the tree is balanced
- Storage is linear



Metric Trees



Metric Trees

NN query on a metric tree:

- Given q, traverse the tree, update the minimum $d_{\min}$ of the distances of q to the traversed ball centres, and eliminate any subtree whose ball of centre p and radius R satisfies

$$|R - D(p, q)| > d_{\min}$$

  - The elimination follows from Triangle Inequality Bound (3) – all sites in the subtree must be more than $d_{\min}$ away from q

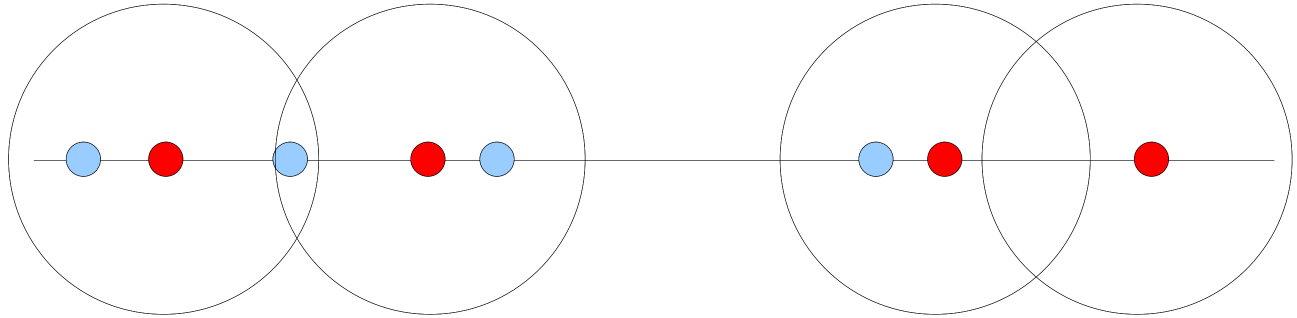
Dimension

What is "dimension"?

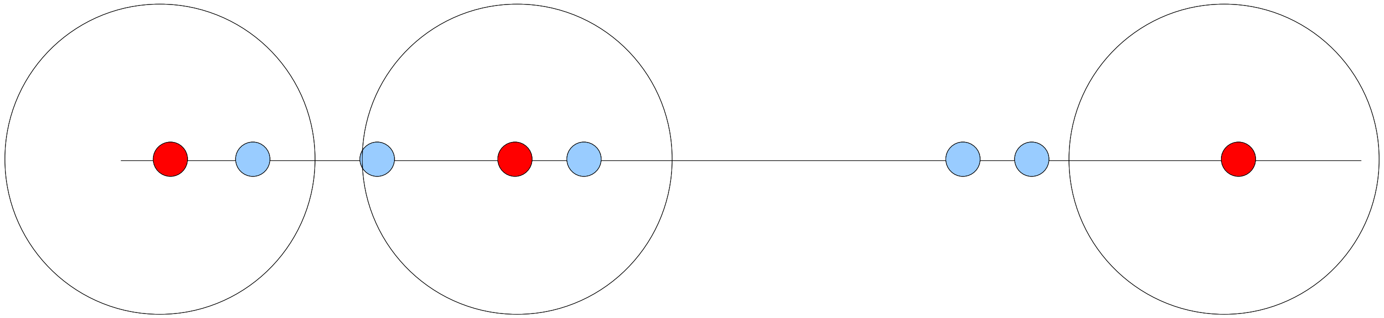
- A way of assigning a real number d to a metric space Z
- Generally "intrinsic", i.e. the dimension depends on the space Z itself and not on any larger space in which it is embedded
- Many different definitions
  - Box dimension
  - Hausdorff dimension
  - Packing dimension
  - Doubling dimension
  - Renyi dimension
  - Pointwise dimension

Coverings and Packings

- Given: Bounded metric space $Z := (U, d)$
- An **ε-cover** of Z is a set $Y \subset U$ s.t. for every $x \in U$, there is some $y \in Y$ with $D(x, y) < \varepsilon$



- A subset Y of U is an **ε-packing** iff $D(x, y) > 2\varepsilon$ for every pair $x, y \in Y$



Coverings and Packings

- Covering number $\mathrm{C}(U, \varepsilon)$: size of smallest ε-covering
- Packing number $\mathrm{P}(U, \varepsilon)$: size of largest ε-packing
- Relation between them:

$$\mathrm{P}(U, \varepsilon) \;\leq\; \mathrm{C}(U, \varepsilon) \;\leq\; \mathrm{P}(U, \varepsilon / 2)$$

  – **Proof:** A maximal $(\varepsilon / 2)$-packing is an ε-cover. Also, for any given ε-cover Y and ε-packing P, every $p \in P$ must be in an ε-ball centred at some $y \in Y$, but no two $p, p' \in P$ can be in the same such ball (else $D(p, p') < 2\varepsilon$ by the Triangle Inequality). So $|P| \leq |Y|$.

- An **ε-net** is a set $Y \subset U$ that is both an ε-cover and an $(\varepsilon / 2)$-packing

Various Dimensions

- Box dimension $\dim_B$: d satisfying $C(U, \varepsilon) = 1 / \varepsilon^d$ as $\varepsilon \to 0$
- Hausdorff dimension $\dim_H$: "critical value" of Hausdorff t-measure $\inf\{\Sigma_{B \in E} \operatorname{diam}(B)^t \mid E \text{ is an } \varepsilon\text{-cover of } U\}$
  - Here ε-cover is generalized to mean a collection of balls, each of diameter at most ε, that cover U
  - Critical value is the t above which the t-measure goes to 0 as $\varepsilon \to 0$, and below which it goes to ∞
- Packing dimension $\dim_P$: Same as Hausdorff but with packing replacing cover and sup replacing inf

Various Dimensions

- Doubling dimension doub_A: Smallest d s.t. any ball $B(x, 2r)$ is contained in the union of at most 2^d balls of radius r
  - Related to Assouad dimension $\dim_A$: d satisfying

    $$\sup_{x \in U, r > 0} C(B(x, r), \varepsilon r) = 1 / \varepsilon^{\mathrm{d}}$$
  - $\dim_A(Z) \leq \mathrm{doub}_A(Z)$
- Doubling measure doub_M: Smallest d satisfying

  $$\mu(B(x, 2r)) \leq \mu(B(x, r))\, 2^d$$

  for a metric space with measure μ
- Pointwise (local) dimension $\alpha_\mu(x)$: For $x \in U$, d s.t.

  $$\mu(B(x, \varepsilon)) = \varepsilon^d \text{ as } \varepsilon \longrightarrow 0$$

Dimension Estimation using NN: An Example

- Given: sample of size n
- The pointwise dimension at x almost surely satisfies

$$\alpha_\mu(x) = \lim_{n \to \infty} \log(k / n) \,/\, \log \delta_{k:n}(x)$$

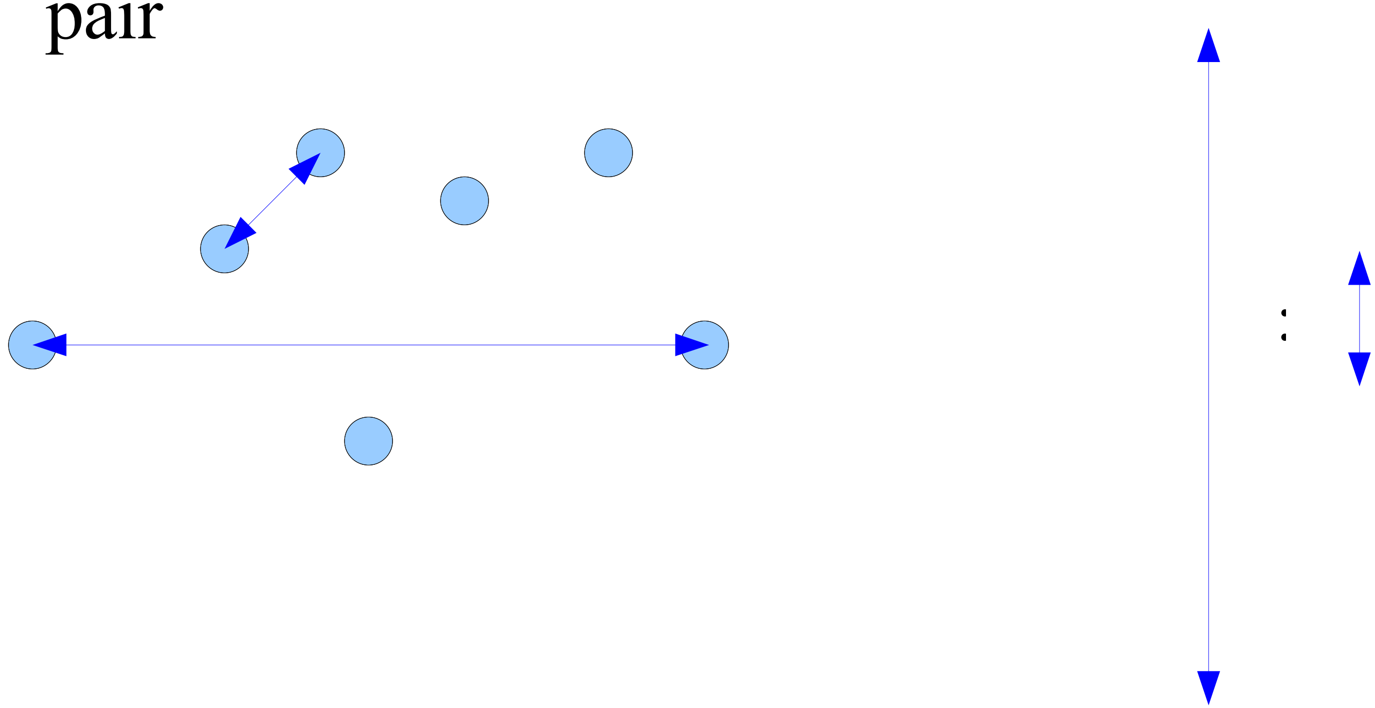
where $\delta_{k:n}(x)$ is the distance of x to its k^{th} nearest neighbour in the sample

- In other words:

$$\delta_{1:n}(x) = n^{-1 / \alpha_\mu(x)}$$

NN in Constant Dimension

- We will first consider spaces (S, D) of constant doubling dimension/measure and bounded spread
  - Spread $\Delta(S)$ is the ratio of the distance between the farthest pair of sites to the distance between the closest pair



A Basic Lemma

- Definition: A site a is k^{th} (γ)-nearest to a site b w.r.t. S if there are at most $k-1$ sites in S whose distance to b is within a factor of γ of the distance of the nearest to b in $S \setminus \{b\}$
- Lemma: For a metric space $Z = (S, D)$ with doubling dimension d, and any site $s \in S$, the number of sites $s' \in S$ for which s is k^{th} (γ)-near in S to s' is $O((2\gamma)^d\, k \log \Delta(S))$, as $1/\gamma \to 0$

Proof of Lemma

- Consider $k = 1$ and a ball $B(s, 2r)$ for some $r > 0$
- There is an (r / γ)-cover Y of $B(s, 2r)$ of size $O((2\gamma)^d)$
- Every site $s' \notin Y$, with $r < D(s, s') \leq 2r$ has a site in Y within distance (r / γ) of it

  $\Rightarrow$ s is not a (γ)-nearest neighbour of s'

  $\Rightarrow$ only points in Y can have s as a (γ)-nearest neighbour

  $\Rightarrow$ the number of sites s' with $r < D(s, s') \leq 2r$ that have s as a (γ)-nearest neighbour is at most $|Y| = O((2\gamma)^d)$
- If p is closest in S to s, at distance r', then consider $r = 2r'$, $4r'$, $8r'$,... At most $\log(\Delta S)$ values of r need be considered, each contributing at most $O((2\gamma)^d)$ sites with s (γ)-near.

Proof of Lemma

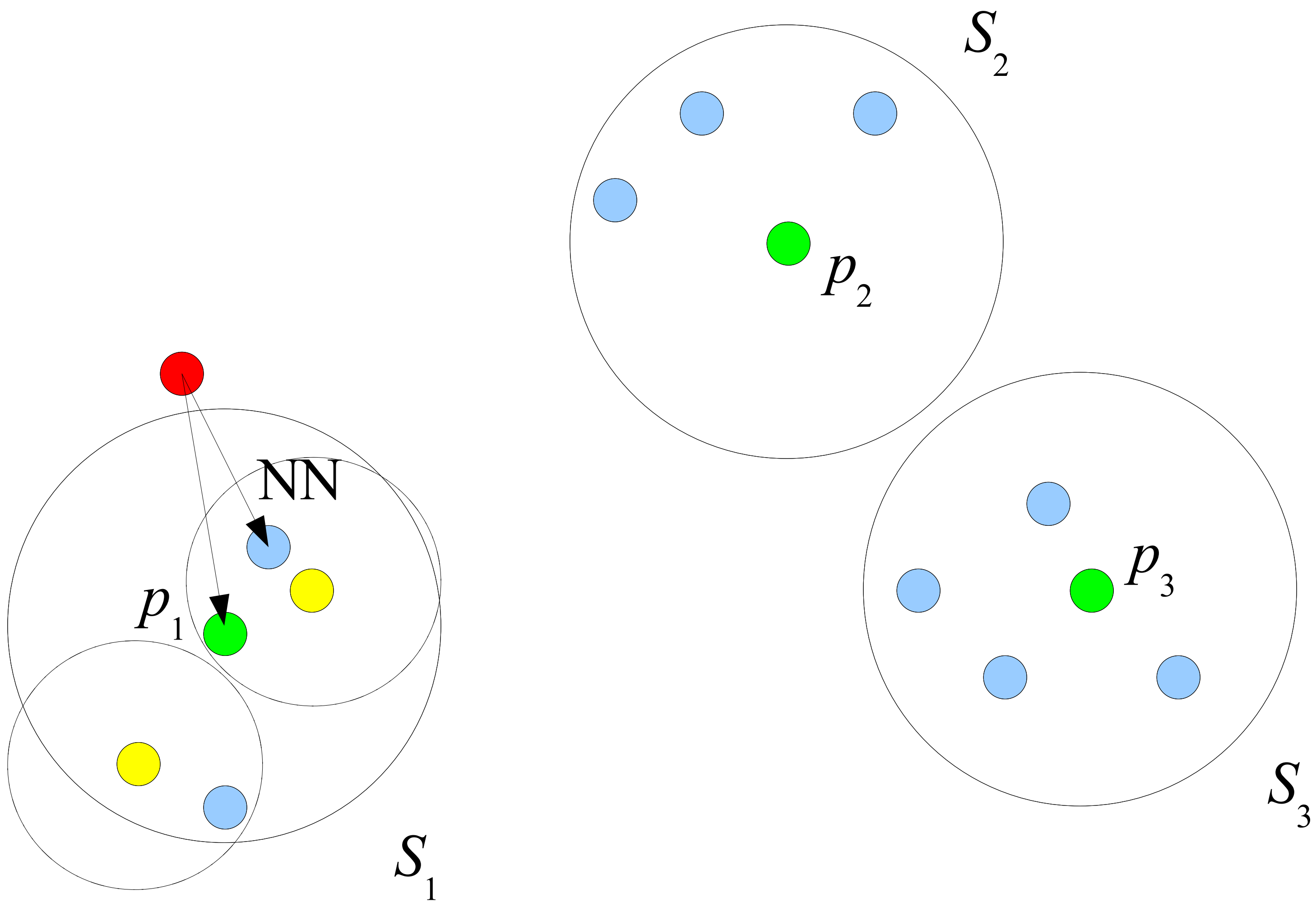
- For $k = 2$, remove all sites of covers in construction for $k = 1$ from S
  - Leaves a metric space with same doubling dimension
- Repeat the previous construction on the remaining sites
  - Gives $O((2\gamma)^d \log \Delta(S))$ new sites with s as a 2^{nd} (γ)-nearest neighbour
- For $k > 2$, repeat this procedure k times

Q.E.D.

Divide-and-Conquer NN

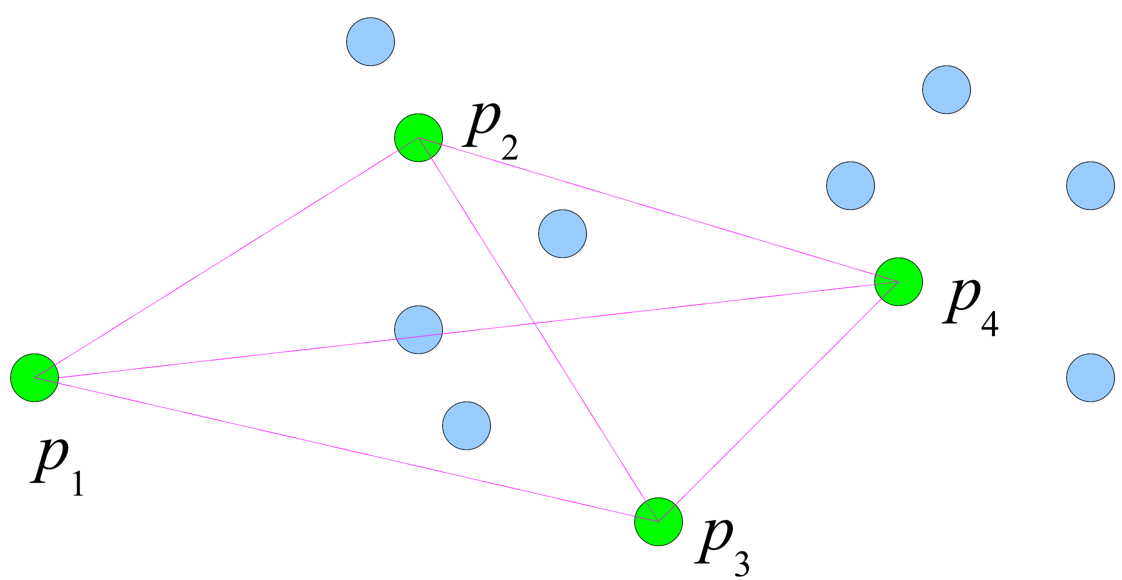
- Idea:
  - Break S into subsets S_1, S_2, S_3,...
  - Characterize each subset by a representative site
  - Use the distances of the query point to the representatives to locate a subset S containing the nearest neighbour
  - Recurse within the subset S_i
- Typically, the set of representatives will be denoted P
- We'll look at spaces with:
  - Constant doubling dimension
  - Constant doubling measure
  - Constant doubling dimension *and* exchangeable queries

Divide-and-Conquer NN



NN in Constant Doubling Dimension

- Metric space $Z = (U, D)$ with $d := \mathrm{doub}_A(Z)$, sites $S \subset U$
- Scale S to fit in ball of radius 1
- Take P to be a δ^2-net for some $\delta > 0$
  - By doubling condition, P has at most $O(1 / \delta^{2d})$ sites

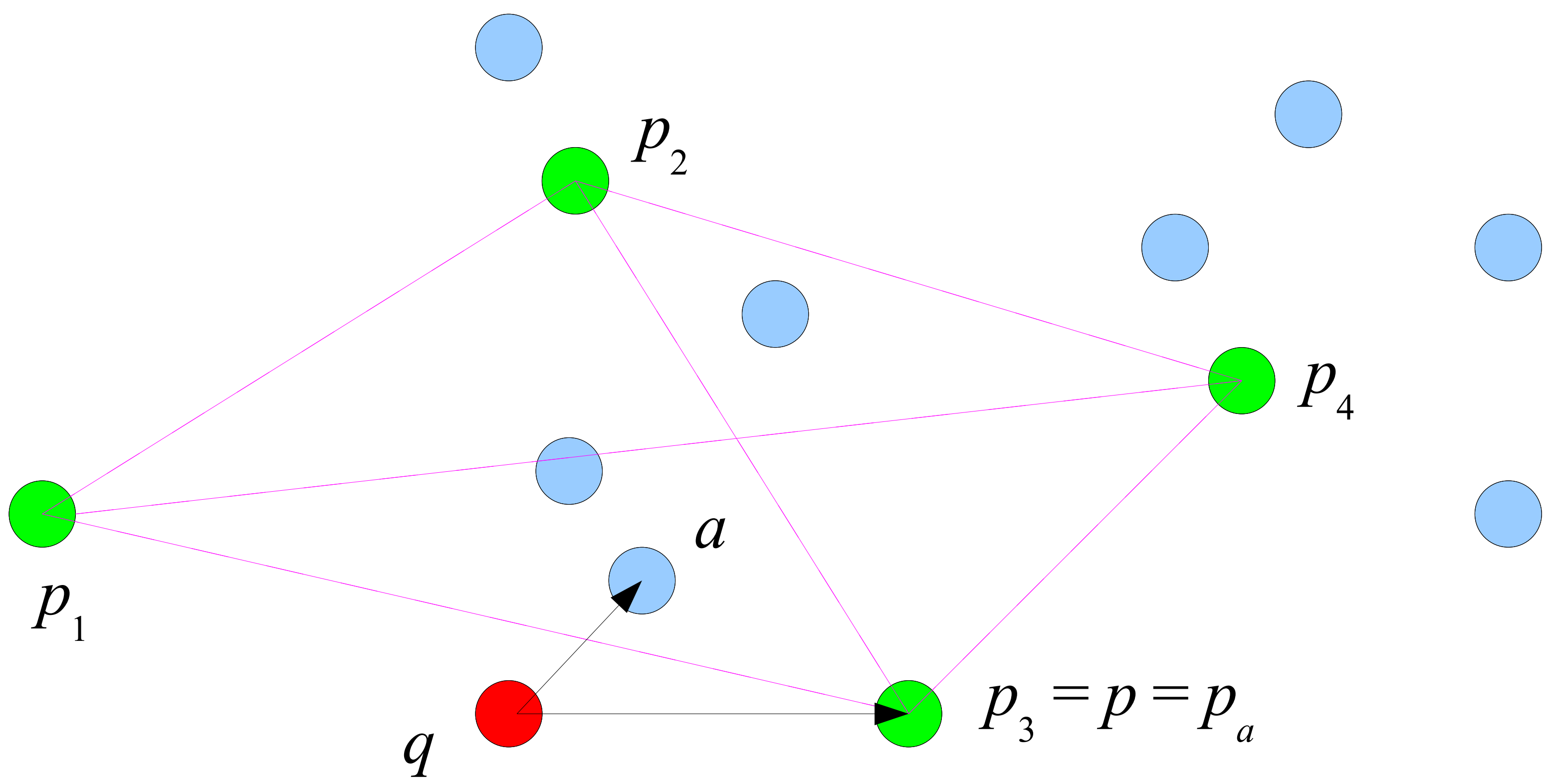


NN in Constant Doubling Dimension

- Suppose q has
  - p as nearest neighbour in P
  - a as nearest neighbour in S
- Suppose p_a is the nearest neighbour of a in P

  $\Rightarrow D(a, p_a) \leq \delta^2$
- $D(q, p) \leq D(q, p_a) \leq D(q, a) + D(a, p_a) \leq D(q, a) + \delta^2$
- If $D(q, a) > \delta$, then p is $(1 + \delta)$-near to q in S
- Else $D(p, a) \leq D(p, q) + D(q, a) \leq 2\delta + \delta^2 \leq 3\delta$ (for $\delta < 1$)

NN in Constant Doubling Dimension

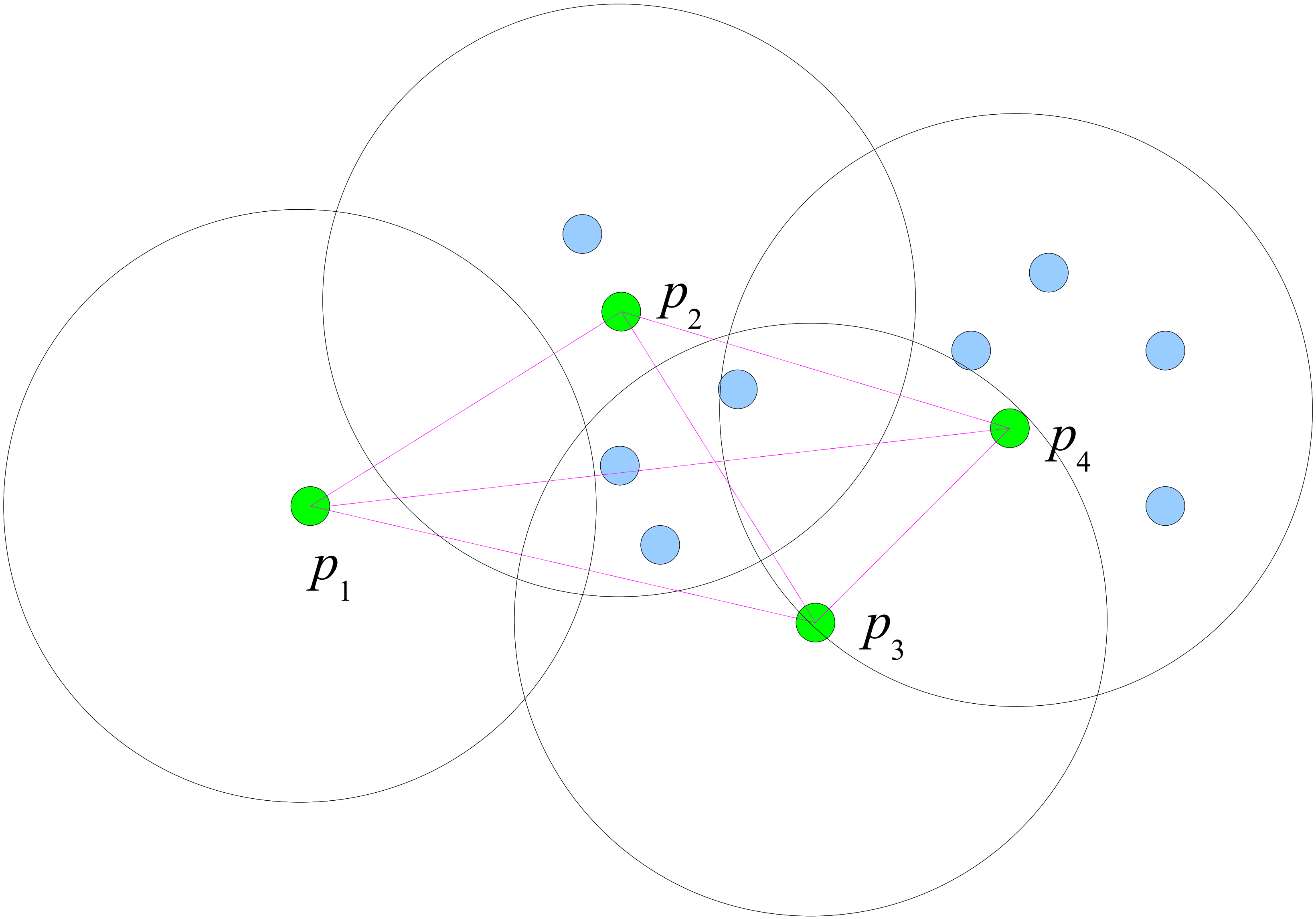


NN in Constant Doubling Dimension

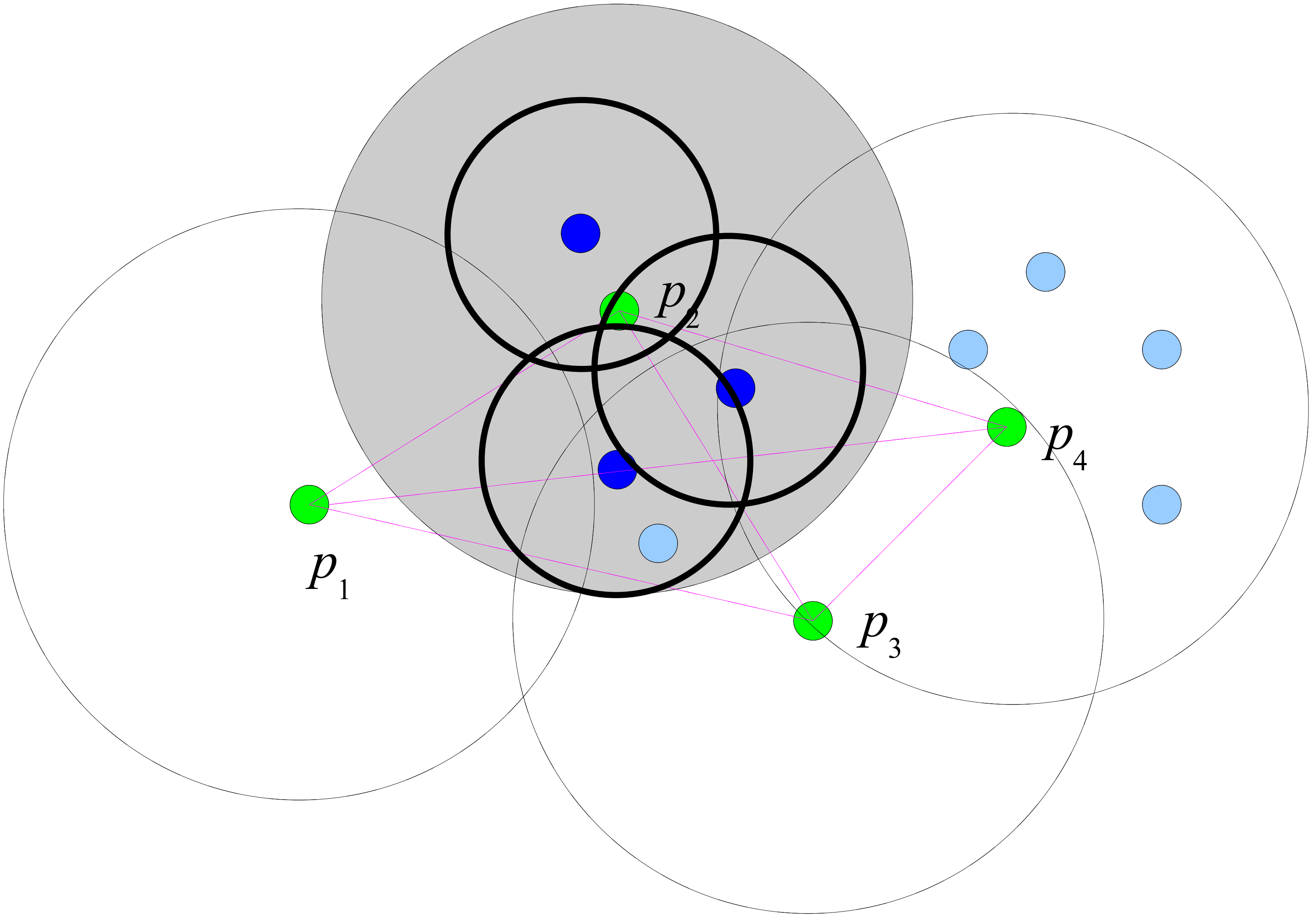
- If p is not the required approximate nearest neighbour, we must have a solution in $B_p := B(p, 3\delta)$
- Recursively build a data structure as follows:
  - For each $p \in P$, construct $S_p := S \cap B_p$
  - Rescale each S_p, construct an δ^2-net for it and recurse
- Assume $\delta < 1/6$. Because of the rescaling, at depth t the (unscaled) sites are in a ball of radius $1/2^t$
- The depth of the tree is $\log(\Delta S)$
- Queries can be answered in $2^{O(d)} \log \Delta(S)$ time

  (assuming δ and hence $|P|$ are constants)

NN in Constant Doubling Dimension



NN in Constant Doubling Dimension



NN in Constant Doubling Measure

- Metric space $Z = (U, D)$ with $d := \text{doub}_{\text{M}}(Z)$, sites $S \subset U$
  - Recall: $\mu(B(x, 2r)) \leq \mu(B(x, r))\, 2^d$
- Let P be a random subset of S, obtained by choosing each site of S independently with probability m / n
  - The expected size of P is m
- For $p \in P$, consider ε_p s.t. $|S \cap B(p, \varepsilon_p)| = K\, n\, (\log n) / m$
- Let p be nearest to q in P
  - If $D(q, p) \leq \varepsilon_p/2$, $B(p, \varepsilon_p)$ contains the NN of q
  - Else, if $\beta := D(q, p)$:
    - $|S \cap B(q, \beta)| \geq |S \cap B(q, 3\beta)| / 4^d \geq |S \cap B(p, \varepsilon_p)| / 4^d = K\, n\, (\log n) / m\, 4^d$

NN in Constant Doubling Measure

- $\Pr[D(q, p) > \varepsilon_p/2$ and p is nearest neighbour of q in $P]$
  $\leq \Pr[B(\mathrm{q}, \beta)$ has no points of $P]$
  $\leq (1 - m / n)^{K\, n\, (\log n) / m\, 4^d}$
  $\leq 1 / n^{K / 4^d}$
- Hence with high probability (at least $1 - 1 / n^{K / 4^d}$), the nearest neighbour of q is in $B(p, \varepsilon_p)$, where p is the nearest neighbour of q in P
- Data structure:
  - Randomly pick P (m := O(log n) is good), and then construct a ball containing $K\, n\, (\log n) / m$ sites around each $p \in P$
  - Recurse in each ball

NN in Constant Doubling Dimension with Exchangeable Queries

We have an approximation algorithm for constant doubling *dimension* and an exact algorithm (with some prob. of error) for constant doubling *measure*

- But the former seems more robust than the latter
- Is there an exact algorithm?
- Yes! If we assume "exchangeability":
  - Sites and queries drawn from same distribution
- Use the usual divide-and-conquer method, using the results of the next slide to construct subsets at each step

NN in Constant Doubling Dimension with Exchangeable Queries

- Pick a random subset $P \subset S$ of size m
- Pick a random subset $P' \subset S$ of size Km
- For each $p \in P$, let $q_p \in P'$ have p nearest in P, but be farthest away among all such sites in P'
- **Lemma 1:** If q is an exchangeable query point with p nearest in P, then with probability $1 - 1/K$, the nearest neighbour to q in S is contained in $B_p := B(q_p, 3D(p, q_p))$
- **Lemma 2:** The expected number of sites in B_p is $2^{O(d)}(Kn/m) \log^2 \Delta(S)$

$M(S, Q)$

- [Clarkson '99]
- A skiplist-type data structure for NN
- Requires auxiliary set Q of m points
- Achieves:
  - Near-linear preprocessing and storage
  - Sublinear query time
- Analysis requires:
  - Exchangeability of q, Q and S
    - Q is "typical set of queries"

$M(S, Q)$

- Definition:
  - $p \in S$ is a (γ)-nearest neighbour of q w.r.t. $R \subset S$ if $D(p, q) \leq \gamma D(q, R)$
  - Denote this by $q \xrightarrow{\gamma} p$ or $p \xleftarrow{\gamma} q$
- Pick γ, and construct $M(S, Q)$ as follows:
  - Let $(p_1, p_2, \ldots, p_n)$ be a random permutation of S
  - Let $R_i := \{p_1, p_2, \ldots, p_i\}$
  - Similarly shuffle Q
    - Q_j is a random subset of Q of size j
    - Define $Q_j := Q$ for $j > m$

$M(S, Q)$

- Define A_j as:

  $\{p_i \mid i > j, \exists\, q \in Q_{Ki} \text{ with } p_j \overset{1}{\leftarrow} q \overset{\gamma}{\rightarrow} p_i, \text{ w.r.t. } R_{i-1}\}$

  - p_j is the nearest neighbour of q in R_{i-1}
  - $D(q, p_i) \le \gamma D(q, p_j)$
  - Construction can be done by adding random points (without repetition) from S one at a time to construct R_i from R_{i-1}, and updating A_j's at each step
- The sites in A_j are in increasing order of index i
- Searching is exactly the same as for Orchard's Algorithm, with p_1 the initial candidate

$M(S, Q)$: Failure Probability

- Assume Q and q are exchangeable and $Kn < m = n^{O(1)}$. If $\gamma = 3$, the probability that $M(S, Q)$ fails to return a nearest site to q in S is $O(\log^2 n) / K$
  - Holds in *any* metric space
- For general γ: Suppose $Z := (U, D)$ has a "γ-dominator bound". Under the same conditions as above, but for general γ, the failure probability is $O(D_\gamma \log^2 n) / K$

$M(S, Q)$: Failure Probability

- γ-dominator bound:
  - Let $R \subset U$. The "nearest neighbour ball" $B(q, R)$ of q w.r.t. R is the set of all points in U closer to q than to any (other) point in R
  - Let $C(p, R)$ be the union of balls $B(q, R)$ over all potential query points q with p closest in R
    - The union is over the Voronoi cell Vor(p) of p w.r.t. R
  - Approximate $C(p, R)$ by $C_\gamma'(p, R)$:
    - Take the union only over $Q \cap$ Vor(p)
    - Expand each ball by a factor γ
  - U has a γ-dominator bound if for every p, R, $\exists$ finite Q of size at most D_γ s.t. $C(p, R) \subset C_\gamma'(p, R)$

$M(S, Q)$: Query Time

- Z has a nearest neighbour bound if $\exists$ a constant N s.t. for all $a \in U$ and any $W \subset U$, the number of $b \in W$ s.t. a is a nearest neighbour of b w.r.t. to W is at most N
- $v(x, W) := \max\{D(x, y) \mid x \in R\} \,/\, D(x, R)$
- $N_\gamma(x, W) :=$ points of W for which x is a (γ)-nearest neighbour w.r.t. W
- $N_{\gamma,v} := \max\{|N_\gamma(x, W)| : x \in U, W \subset U, v(x, W) \leq v\}$
  - ... if it exists
  - $\gamma \geq 1$, $v > 0$

$M(S, Q)$: Query Time

- Z has a γ-nearest neighbour bound if it has a nearest neighbour bound *and* $N_{\gamma,v}$ exists for every $v > 0$
- If S, Q and q are *all* exchangeable and Z has a γ-nearest neighbour bound, M(S, Q) returns an answer in time

$$\mathrm{O}(N_{\gamma,\Delta(S \cup Q)} N_1 K \log n)$$

$M(S, Q)$: Storage

- Z has a sphere-packing bound if for any real number ρ, $\exists$ an integer constant S_ρ s.t. for all $a \in U$ and $W \subset V$, if $|W| > S_\rho$ and $D(w, a) \leq C$ for all $w \in W$ for some C, then $\exists$ w, w' s.t. $D(w, w') < C / \rho$
- Sphere-packing bounds imply the other two bounds
- If Q and S are exchangeable and Z has a sphere-packing bound, $M(S, Q)$ uses $O(S_{2\gamma} \log \Delta(S \cup Q))\, Kn$ expected space