Approximate Nearest Neighbor via Point-Location among Balls
Outline

• Problem and Motivation
• Related Work
• Background Techniques
• Method of Har-Peled (in notes)
Problem

- $P$ is a set of points in a metric space.
- Build a data structure to efficiently search $ANN$.
Motivation

• Nearest Neighbor Search has lots of applications.

• Curse of dimensionality
  - Voronoi diagram method exponential in dimension.

• Settle for approximate answers.
Related Work

- Indyk and Motwani
- Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality
- Reduced ANN to Approximate Point-Location among Equal Balls.
- Polynomial construction time.
- Sublinear query time.
Related Work

- Har-Peled

- A Replacement for Voronoi Diagrams of Near Linear Size

- Simplified and improved Indyk-Motwani reduction.

  - Better construction and query time.
Related Work

- Sabharwal, Sharma and Sen

- Nearest Neighbors Search using Point Location in Balls with applications to approximate Voronoi Decompositions.

- Improved number of balls by a logarithmic factor.

- Also a complex construction which only requires $O(n)$ balls.
Metric Spaces

- Pair \((X, d)\)
- \(d: X \times X \rightarrow [0, \infty)\)
- \(d(x, y) = 0\) iff \(x = y\)
- \(d(x, y) = d(y, x)\)
- \(d(x, y) + d(y, z) \geq d(x, z)\)
Hierarchically well-separated Tree (HST)

- Each vertex $u$ has a label $\Delta_u \geq 0$.
- $\Delta_u = 0$ iff $u$ is a leaf.
- If a vertex $u$ is a child of a vertex $v$, then $\Delta_u \leq \Delta_v$.
- Distance between two leaves $u, v$ is defined as $\Delta_{lca(u,v)}$ where $lca$ is the least common ancestor.
Hierarchically well-separated Tree (HST)

- Each vertex $u$ has a representative descendant leaf $\text{rep}_u$.
- $\text{rep}_u \in \{\text{rep}_v \mid v \text{ is a child of } u\}$.
- If $u$ is a leaf, then $\text{rep}_u = u$. 
Metric $t$-approximation

- A metric $N$ $t$-approximates a metric $M$, if they are on the same set of points, and $d_M(x,y) \leq d_N(x,y) \leq td_M(x,y)$ for any points $x,y$. 

\[ d_M(x,y) \]
\[ d_N(x,y) \]
Any n-point metric is $2^{(n-1)}$-approximated by some HST.
First Step: Compute a 2-spanner

- Given a metric space $M$, a 2-spanner is a weighted graph $G$ whose vertices are the point of $M$ and whose shortest path metric 2-approximates $M$.

- $d_M(x,y) \leq d_G(x,y) \leq 2d_M(x,y)$ for all $x,y$.

- Can be computed in $O(n \log n)$ time — Details in Chapter 4.
Construct a HST which \((n-1)\)-approximates the 2-spanner

- Compute the minimum spanning tree of \(G\), the 2-spanner
Construct a HST which \((n-1)\)-approximates the 2-spanner

- Construct the HST using a variation of Kruskal’s algorithm
- Order the edges in non-decreasing order.
Construct a HST which (n-1)-approximates the 2-spanner.

- Start with n 1-element HSTs.
Construct a HST which \((n-1)\)-approximates the 2-spanner.

- Add the edges one by one, and merge corresponding HSTs by adding a parent node with \(\Delta\) label equal to \((n-1)\) times the edge’s weight.
Construct a HST which \((n-1)\)-approximates the 2-spanner.

- Add the edges one by one, and merge corresponding HSTs by adding a parent node with \(\Delta\) label equal to \((n-1)\) times the edge’s weight.
Construct a HST which \((n-1)\)-approximates the 2-spanner

- Add the edges one by one, and merge corresponding HSTs by adding a parent node with \(\Delta\) label equal to \((n-1)\) times the edge’s weight.
Construct a HST which \((n-1)\)-approximates the 2-spanner.

- Add the edges one by one, and merge corresponding HSTs by adding a parent node with \(\Delta\) label equal to \((n-1)\) times the edge’s weight.
Construct a HST which $(n-1)$-approximates the 2-spanner.

- Add the edges one by one, and merge corresponding HSTs by adding a parent node with $\Delta$ label equal to $(n-1)$ times the edge’s weight.
The HST \((n-1)\)-approximates the 2-spanner

Consider vertices \(x\) and \(y\) in the graph and the first edge \(e\) that connects their respective connected components.
The HST \((n-1)\)-approximates the 2-spanner

- Let \(C\) be the connected component containing \(x\) and \(y\) after \(e\) is added.

- \(w(e) \leq d_G(x,y) \leq (|C|-1)w(e) \leq (n-1)w(e) = d_H(x,y)\)

- \(d_G(x,y) \leq d_H(x,y) \leq (n-1)\)

\[d_G(x,y)\]
Any $n$-point metric is $2^{(n-1)}$-approximated by some HST.
Target Balls

- Let $B$ be a set of balls such that the union of the balls in $B$ contains the metric space $M$.

- For a point $q$ in $M$, the target ball of $q$ in $B$, denoted $⊙_B(q)$, is the smallest ball in $B$ that contains $q$.

- We want to reduce ANN to target ball queries.
A Trivial Result — Using Balls to Find ANN

- Let $B(P,r)$ be the set of balls of radius $r$ around each point $p$ in $P$.

- Let $B$ be the union of $B(P, (1+\varepsilon)^i)$ where $i$ ranges from $-\infty$ to $\infty$.

- For a point $q$, let $p$ be the center of $b = \odot_B(q)$. Then $p$ is $(1+\varepsilon)$-ANN to $q$. 
A Trivial Result — Using Balls to Find ANN

- Let $s$ be the nearest neighbor to $q$ in $P$.
- Let $r = d(s,q)$.
- Fix $i$ such that $(1+\varepsilon)^i < r \leq (1+\varepsilon)^{i+1}$
- Radius of $b > (1+\varepsilon)^i$
- $d(s,q) \leq d(p,q) \leq (1+\varepsilon)^{i+1} \leq (1+\varepsilon)d(s,q)$
What We Need to Fix

• This works, but has unbounded complexity.
• We want the number of balls we need to check to be linear.
• We first try limiting the range of the radii of the balls.
• First, we need to figure out how to handle a range of distances.
Near-Neighbor Data Structure (NNbr)

• Let $d(q,P)$ be the infinum of $d(q,p)$ for $p \in P$.

• NNbr(P,r) is a data structure, such that when given a query point q, it can decide if $d(q,P) \leq r$.

• If $d(q,P) \leq r$, NNbr(P,r) also returns a witness point $p$ such that $d(q,p) \leq r$. 
Near-Neighbor Data Structure (NNbr)

- Can be realized by \( n \) balls of radius \( r \) around the points of \( P \).
- Perform target ball queries on this set of balls.
Interval Near-Neighbor Data Structure

- NNbr data structure with exponential jumps in range.

- $N_i = \text{NNbr}(P, (1+\varepsilon)^i a)$

- $M = \log_{1+\varepsilon}(b/a)$

- $I(P,a,b,\varepsilon) = \{N_0, ..., N_M\}$
Interval Near-Neighbor Data Structure

- \( \log_{1+\varepsilon}(b/a) = \Theta(\log(b/a)/\log(1+\varepsilon)) = \Theta(\varepsilon^{-1}\log(b/a)) \) NNbr data structures.

- \( \Theta(\varepsilon^{-1}n\log(b/a)) \) balls.
Using Interval NNbr to find ANN

• First check boundaries: $O(1)$ NNbr queries, $O(n)$ target ball queries.

• Then, do binary search on the $M$ NNbr’s. This is $O(\log(\varepsilon^{-1}\log(b/a)))$ NNbr queries, or $O(n\log(\varepsilon^{-1}\log(b/a)))$ target ball queries.

• Fast if $b/a$ small.
Faraway Clusters of Points

- Let $Q$ be a set of $m$ points.
- Let $U$ be the union of the balls of radius $r$ around the points of $Q$.
- Suppose $U$ is connected.
Faraway Clusters of Points

- Any two points $p, q$ in $Q$ are in distance $\leq 2r(m-1)$ from each other.

- If $d(q,Q) > 2mr/\delta$, any point of $Q$ is a $(1+\delta)$-ANN of $q$ in $Q$. 
Faraway Clusters of Points

- Let $s$ be the closest point in $Q$ to $q$.
- Let $p$ be any member of $Q$.
- $2mr/\delta < d(q,s) \leq d(q,p) \leq d(q,s) + d(s,p) \leq d(q,s) + 2mr \leq (1+\delta)d(q,s)$.