Proximity/Nearest Neighbors

Problem 1: Given \(n\) red pts \(P\) & \(n\) blue pts \(Q\) in \(\mathbb{R}^d\), find bi-chromatic closest pair.

Problem 2: Find nearest blue neighbor for every red pt.

Known exact alg's:
- \(d=2\) \(O(n \log n)\)
- \(d=3\) \(O(n^{4/3})\)
- \(d > 3\) \(O(n^{2 - \frac{d}{2(d-1)} + \epsilon})\)

Method 0 - rounding dirs:
  - For each red pt \(q\), unit vector
  - For each dir \(v \in V_q\), find blue pt \(q'\) in cone \(\{ \mathbf{p} : \angle (p-q, u) < \theta \}\) with smallest \(p \cdot u\)

Analysis:
- \(|p - q| \geq (p-q) \cdot u \geq |p-q| \cos \theta \sim (1 - O(\theta^2)) |p-q|

Set \(\theta = \sqrt{3}\)

\[O\left(\frac{1}{8d-1}\right) = O\left(\frac{1}{2^{d-1}/2}\right)\] dirs

Not obvious how to do cone range search...
Method 1 - grid
approx decision problem: given fixed r, return some pair of dist \( \leq (1+\epsilon)r \)
or declare closest pair dist > r.

Method 1.1
form grid of side length \( \frac{r}{\epsilon} \)
hash pts to grid cells for each nonempty grid cell \( B \)
check all grid cells \( B' \) with \( d(B,B') \leq r \).

runtime \( O\left(\frac{1}{\epsilon^2} n\right) \)

how to reduce to decision problem?
binary search - extra \( \log n \) factor
random walk ...

better \( \epsilon \)-dependence?

Method 1.2
form grid of side length \( \frac{r}{C} \)
if some grid cell contains both red & blue, done
for each nonempty grid cell \( B \)
for each nonempty blue grid cell \( B' \) with \( d(B,B') \leq r \)
if \( d(B,B') \leq r/2 \), done
if \( d(B,B') > r/2 \)
\( B \) & \( B' \) are well-separated: solve subproblem in \( B \) & \( B' \)
How to solve well-separated case?

build grid $G_8$ of side length $8r$ over middle hyperplane $H$
for each grid pt $g \in G_8$,

$$P_g = \text{nearest neighbor of } g \text{ in } B$$
$$Q_g = \ldots$$

return closest among $P_g, Q_g$

Analysis: say $p^*, q^*$ is closest pair

Let $g$ be closest grid pt to $p^*q^* \cap H$

$$||P_g - Q_g|| \leq ||P_g - g|| + ||g - Q_g||$$
$$\leq ||p^* - q^*|| + ||g - Q^*||$$
$$= \frac{||p^* - q^*||}{\cos \theta}$$

approx factor $1 + \varepsilon$ by setting $\varepsilon \sim \sqrt{E}$

total time $O\left(\frac{1}{(\varepsilon d+1) n}\right)$
**Improvement:**

by discrete Voronoi diagram (similar to $\varepsilon$-kernels)

near $O^*(n + \frac{1}{\varepsilon (d+1)/2})$ time for well-separated case

**Total time**

$$O^* \left( \sum \left( n_i + \frac{1}{\varepsilon (d+1)/2} \right) \right)$$

with $\sum n_i = O(n)$

$$= O^* \left( \frac{n}{\varepsilon (d+1)/2} \right)$$

no better!

**Further Improvement:** (C. '17, Agra-da Fonseca-Mount '17)

$$O^* \left( \sum \min \left\{ n_i + \frac{1}{\varepsilon (d+1)/2}, n_i^{2/3} \right\} \right) \text{ if } n_i \geq \frac{1}{\varepsilon (d+1)/2}$$

$$\leq O^* \left( \sum \frac{1}{\varepsilon (d+1)/4} n_i \right)$$

$$= O^* \left( \frac{n}{\varepsilon (d+1)/4} \right)$$

What if $r$ is not fixed? e.g. nearest neighbor of each red pt

**Method 2 - Quadtrees**

**Idea:** hierarchy of grids

```
+  ->  +  ->  +  ...
```

...
Def: A quadtrees cell \( B \) is a grid cell of side length \( 2^d \) for some \( d \in \{0, 1, \ldots, \log U\} \).

Space \( O(n \log U) \) can be reduced to \( O(n) \) by compressed quadtrees (short-cutting deg-1 nodes).

Decision query \( (B, q, r) \):  // find pt of dist < \((1/4) r\) from \( q \) or declare nearest neighbor dist \( r \)
if ball \( (q, r) \) does not intersect \( B \) return
if \( B \) has side length \( \leq 2r \) return any pt in \( B \)
for each child \( B_i \) of \( B \)
decision query \( (B_i, q, r) \)

\[ \implies \text{query time} = O\left( \# \text{quadtrees cells of side length } > 2r \right) \]

\[ \leq O\left( \sum_{l : 2^l > 2r} \left[ \frac{r}{2^{2l-1}} \right] \right) \]
\[ O\left( \frac{1}{\varepsilon^2} + \frac{1}{(2\varepsilon)^d} + \ldots \right) = O\left( \log U + \frac{1}{\varepsilon^d} \right) \quad \text{(for decs.)} \]

(Rem: for nearest neighbor, do binary search or "Priority" search)

can we reduce \( \log U \) to \( \log n \)?

Method 22: Balanced Quadtrees

Lemma

\[ \exists \text{ quadtree cell } B \]
\[ \text{s.t. } |P \cap B|, |P - B| \leq \frac{2^d}{2^{d+1}} n \]

**Pf:** Let \( B \) be smallest quadtree cell with \( |P \cap B| \geq \frac{1}{2^{d+1}} n \).

Then \( |P - B| \leq \frac{2^d}{2^{d+1}} n \).

For each child \( B_i \) of \( B \),
\[ |P \cap B_i| < \frac{1}{2^{d+1}} \Rightarrow |P \cap B| \leq \frac{2^d}{2^{d+1}} n \]

recorse in \( P \cap B \) & \( P - B \)

\[ \Rightarrow \text{ binary tree of height } \log \frac{n}{2^d} = O(\log n) \]

Query time
\[ O\left( \# \text{ quadtree cells } B \text{ of side length } > \varepsilon \right) \]
\[ \times \log n \]

\[ = O\left( \frac{1}{\varepsilon^2} \log n \right) \]

\[ \leq \sum \text{ a cell may occur in multiple nodes} \]

Ranks Known alternatives:

- BBD trees (Arya, Mount et al. '95)
- BAR trees, fair-split trees, ring-cover trees, skip quadtrees,
Method 2.3: With shifting

naive-query \((B, q)\):
find child \(B_i\) containing \(q\)

naive-query \((B_i, q)\)

When does this work?

Let \(r^*\) = nearest neighbor dist from \(q\).

Def ball \((q, r^*)\) is \(k\)-good if it lies in a quadtree cell
of side length \(<4kr^*\)

Analysis: assuming ball \((q, r^*)\) is \(k\)-good
let \(p\) be output pt.

\[ \|p-q\| < 4k\sqrt{d} r^* \]

\[ O(1) \text{ approx if k const} \]

Query time: \(O(\log U)\)

(can be reduced to \(O(\log \gamma)\)
by binary search in \(k\)-th order)

Shifting Lemma (Rand. Version)
[Bern'93] can make \(v_1 = \ldots = v_d\).

Shift all pts by rand. vector \(v = (v_1, \ldots, v_d) \in (0, 1)^d\).

Then ball \((q, r^*)\) is \(k\)-good with const prob if \(k > d\).

Pf. Say \(2^l \leq 4kr^* \leq 2^{l+1}\).

\[ \Pr[\text{bad}] = \Pr[\text{ball}((q, r^*) \text{ crosses boundary of quadtree cell of side length } 2^l)] \]

\[ \leq d \cdot \frac{2r^*}{2^l} \leq d \cdot \frac{1}{c} = \square \]
Shifting Lemma (Dot. Version) (c'98)

Let \( U = 2^w \). Say \( k \) is odd with \( k > d \).

Define \( k \) vectors \( v(c) = \left( \frac{c2^w}{k}, \ldots, \frac{c2^w}{k} \right) \), \( i = 0, \ldots, k-1 \).

Then \( V \) has \( i \) s.t. it is \( k \)-good after shifting by \( v(c) \).

In fact, for all but \( d \) indices.

**Pf.** Fix \( c, r \). Fix \( i \).

Bad \( \Rightarrow \) for some \( \ell \in \{1, \ldots, d\} \)

\[ \left( \frac{c2^w + i2^w}{k} \right) \mod 2^\ell \neq \left( \frac{r2^\ell}{2^\ell} \right) \]

\[ \Rightarrow \left( \frac{c2^w + i2^w - \ell}{2^\ell} \right) \mod k \]

\[ \neq \left[ \frac{r2^\ell}{2^\ell} - \frac{k}{2^\ell} \right] \]

For some \( \ell \)

\[ \Rightarrow 2^{w+1} i + \left[ \left( \frac{k2^w}{2^\ell} \right) \right] \equiv 0 \mod k \]

(\( ax + b \equiv 0 \mod k \)
has unique soln.
if \( a, k \) rel. prime)

\[ \Rightarrow \] at most \( d \) choices of \( i \). \( \square \)

**Run: const approx \( \rightarrow 1 + \varepsilon \) approx.
query time \( O(\frac{1}{\varepsilon d \log n}) \).**

Can be improved to \( O(\frac{1}{\varepsilon^2 d^2 \log n}) \)

with \( O(\frac{1}{\varepsilon^2 d^4 n}) \) space.

(time/space trade-offs)