Nearest neighbor searching

Preprocess points $P = \{p_1 \ldots p_n\}$

Later, given point q

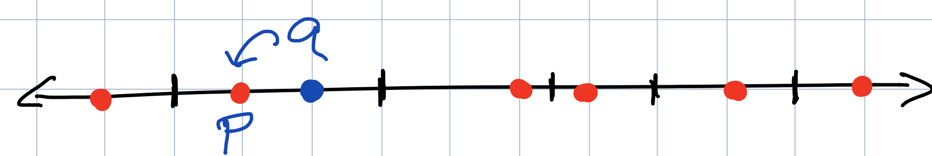
quickly find $\min\{\underbrace{d(p,q)}_{\text{distance}} \mid p \in P\}$

Euclidean distance in $\mathbb{R}^d$

Fast alg/small space when $d \le 2$

Brute force: $O(dn)$ time

d=1:



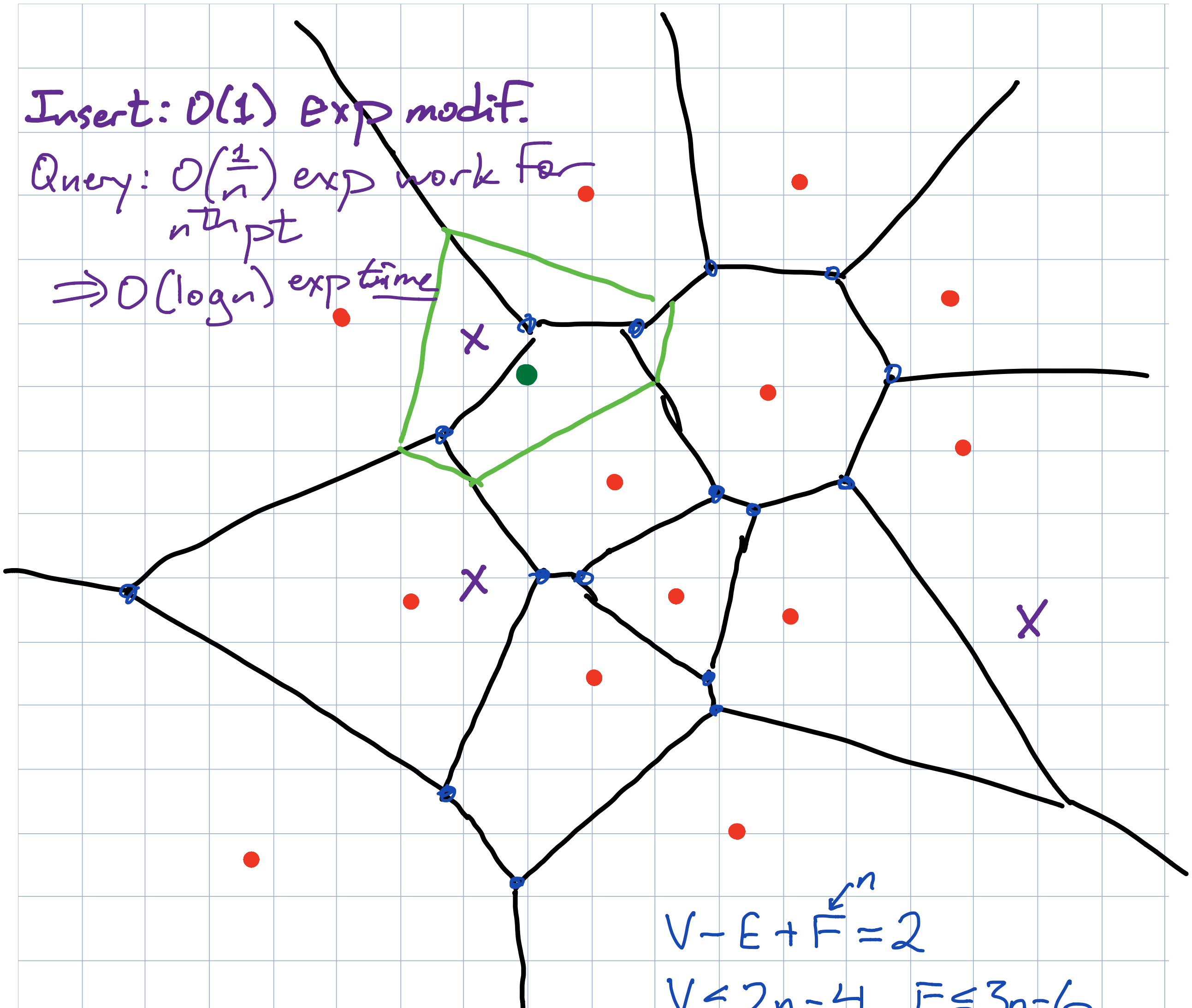
Binary search! $O(\log n)$ time $O(n)$ space

d=2: Voronoi diagram [1906]

Insert: $O(1)$ exp modif.

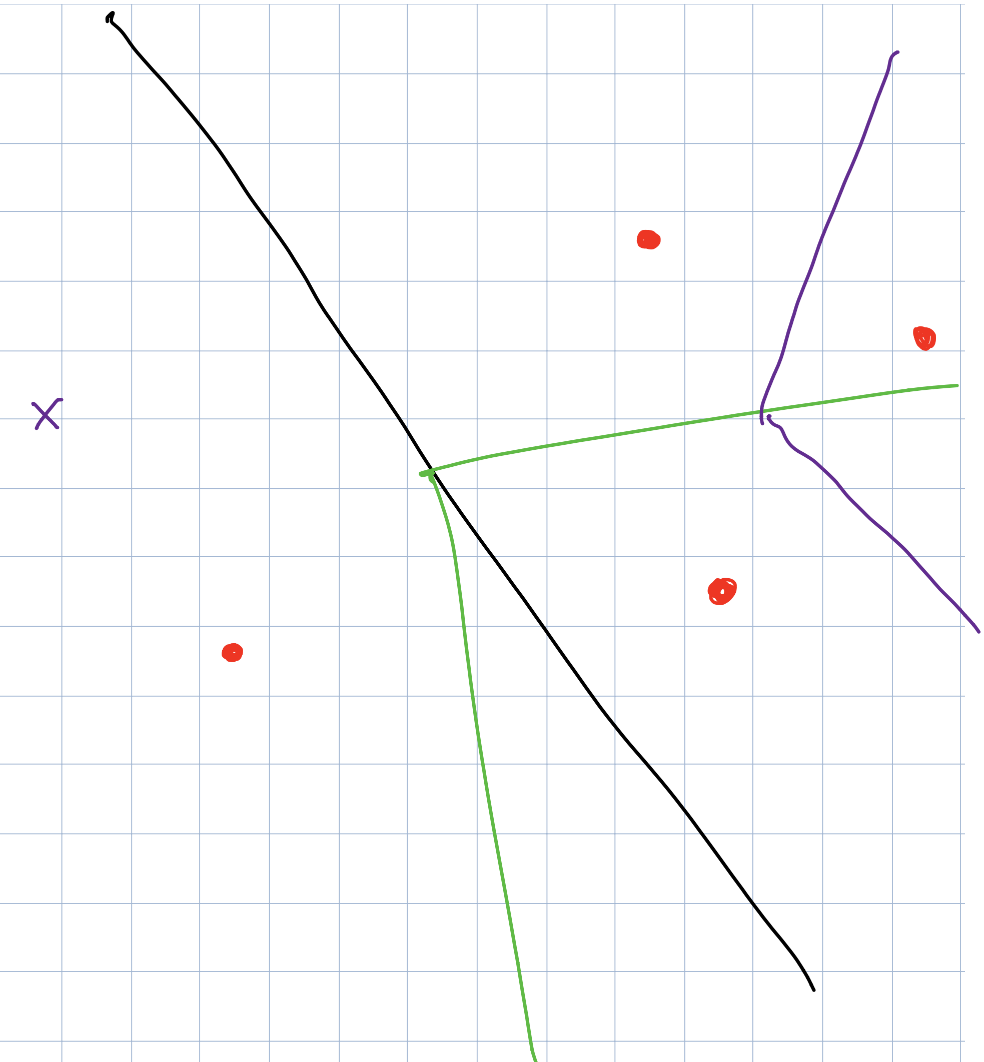
Query: $O(\frac{1}{n})$ exp work for n^{th} pt

$\Rightarrow O(\log n)$ exp time

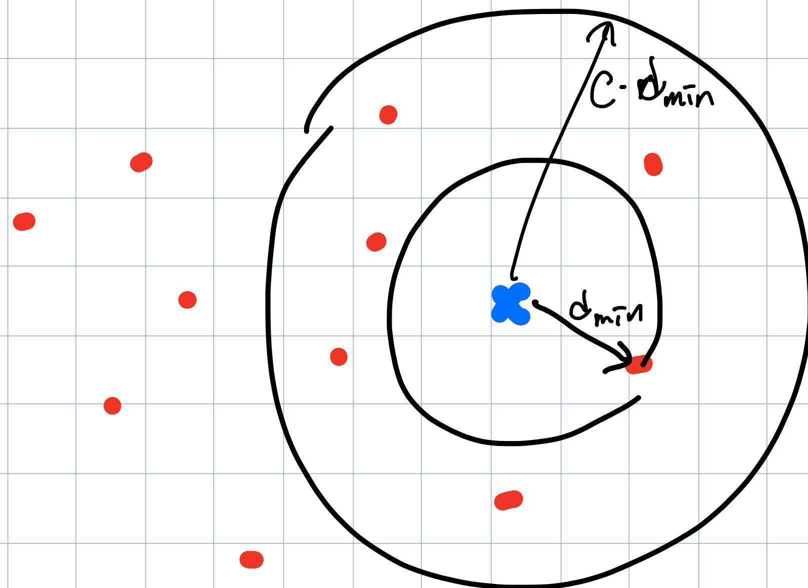


$$V - E + \overset{n}{F} = 2$$

$$V \le 2n - 4 \quad E \le 3n - 6$$



Approx. nearest neighbor query:



Return any point $p_i \in P$ s.t.

$$d(p_i,q) \le c \cdot \min\{d(p,q) \mid p \in P\}$$

with prob $1-\delta$.

<u>Locality-sensitive hashing</u>

[Har-Peled 2001]

<u>Approx. near neighbor search</u>

Given P to preprocess

Given point q, distances r, R

If there is a point $p \in P$ s.t. $d(p,q) \le r$

return a point $p' \in P$ s.t. $d(p',q) \le R$

with prob $1-\delta$.

A family $\mathcal{H}$ of hash functions is locality sensitive if

IF $d(p,q) \le r \implies \Pr[h(p)=h(q)] \ge \Pi$

IF $d(p,q) > R \implies \Pr[h(p)=h(q)] \le \pi$

"Points" are d-bit vectors

$\text{dist}(p,q) = \#1\text{'s in } p \oplus q$ Hamming dist

$p = 0000011101$
$q = 0010011100$

dist = 3

$h_i(p) = p_i = i^{th}$ bit in p $\quad \mathcal{H} = \{h_i \mid 1 \le i \le d\}$

$$\Pr[h(p)=h(q)] = \frac{d - \text{dist}(p,q)}{d} = 1 - \frac{d(p,q)}{d}$$

$\Pi = 1 - \frac{r}{d} \quad \pi = 1 - \frac{R}{d}$

Pick $k \cdot l$ functions $h_{ij} \in \mathcal{H}$ independently
$1 \le i \le k \quad 1 \le j \le l$

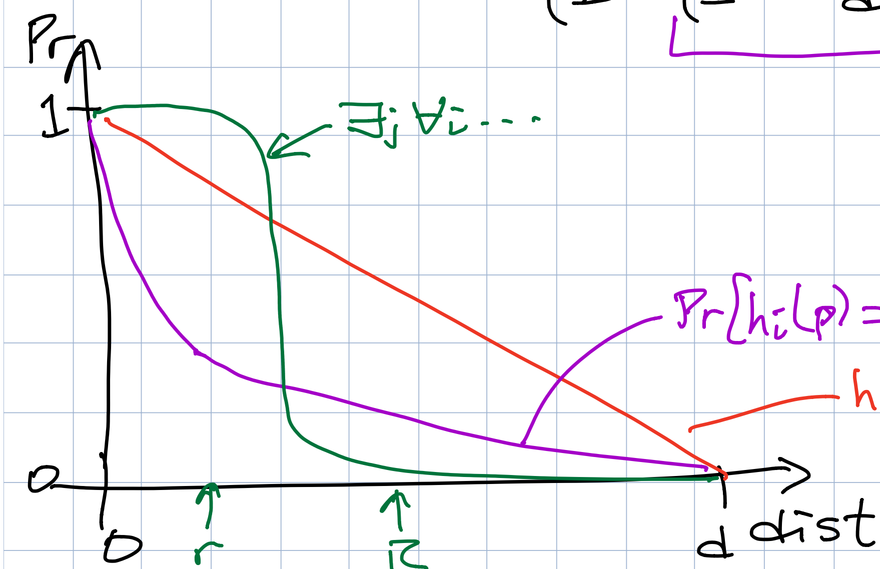
For all j: $h_j(p) = (h_{1j}(p), \ldots, h_{kj}(p)) \in \{0,1\}^k$

Define "collide" = For some j, $h_j(p) = h_j(q)$

$$\boxed{\exists j\ \forall i\ h_{ij}(p)=h_{ij}(q)}$$

$$\Pr[\text{collision}] \leq 1-\left(1-\Pr[h_{ij}(p)=h_{ij}(q)]^{k}\right)^{\ell}$$

$$= 1-\left(1-\left(1-\frac{\text{dist}}{d}\right)^{k}\right)^{\ell}$$



foreach j:
  For each $p \in P$
    store p at $T_j[h_j(p)]$

<u>query(q)</u>
  for each j:
    $S \leftarrow T_j[h_j(q)]$
    check every pt in S
      if close pt found return it
    stop after checking 2ℓ points