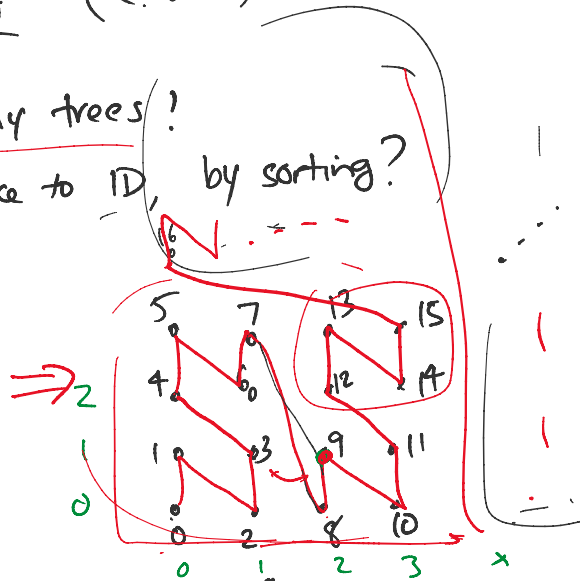
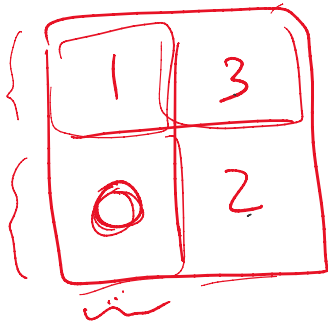


Method 4: Z-Ordering (C'02)

idea - don't use any trees!
directly reduce to 1D, by sorting?



"space-filling curve"
(other exs: Hilbert curve...)

Def Given point $p = (x, y) \in [U]^2$, $U = 2^w$
write $x = a_{w-1} a_{w-2} \dots a_0$ in binary
 $y = b_{w-1} b_{w-2} \dots b_0$
define $\text{shuffle}(p) = a_{w-1} b_{w-1} a_{w-2} b_{w-2} \dots a_0 b_0$

e.g. $p = (2, 1)$
 $x = 2 \rightarrow 010$
 $y = 1 \rightarrow 001$
 $\text{shuffle}(p) \rightarrow 001001 \rightarrow 9$

Obs $\text{shuffle}(p)$ corresponds to position in Z-order
Shuffle maps a quadtree cell
to an interval contiguous ints in 1D. }

Lemma can compare $\text{shuffle}(p)$ with $\text{shuffle}(q)$
in $O(1)$ time.

or $n = (x, y)$ $a = (x', y')$ e.g. $x = 1101$ $y = 1011$
 $\dots$ $\dots$

ff: $p = (x, y), q = (x', y')$ e.g. $x = 1101$ $y = 1011$
 $x' = 1011$ $y' = 1001$
 $x \oplus x' = 0110$ $y \oplus y' = 0010$

$\oplus$ bitwise xor
 msb most significant 1-bit

$\text{shuffle}(p) = 11100111$
 $\text{shuffle}(q) = 11001011$

$\left\{ \begin{array}{l} \text{if } \text{msb}(x \oplus x') \geq \text{msb}(y \oplus y') \\ \text{compare } x \text{ with } x' \\ \text{else compare } y \text{ with } y' \end{array} \right.$

to check $\text{msb}(a) < \text{msb}(b)$:
 check $b < a$ and $b < a \oplus b$.

e.g. $a = 01xxx$ $a = 0001x$ $a = 01xxx$
 $b = 0001x$ $b = 01xxx$ $b = 01xxx$
 $a \oplus b = 01xxx$ $a \oplus b = 01xxx$ $a \oplus b = 00xxx$
 $\min(a, b, a \oplus b) \rightarrow b$ $\rightarrow a$ $\rightarrow a \oplus b$

□

preproc alg'm:

for each of $d+1$ shifts

store pts sorted along z-order

$\Rightarrow O(n \log n)$ time, $O(n)$ space

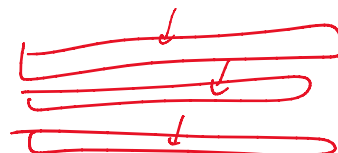
query(q):

for each of $d+1$ shifts,

find prec & succ of q

return closest pt found

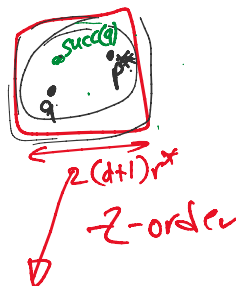
$\Rightarrow O(\log n)$ time



Analysis:

Suppose q is good.

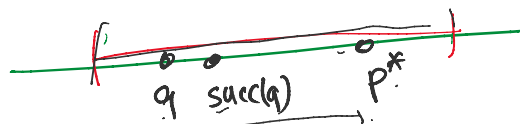
p^* and q are in quadtree cell
of side length $\leq \underline{2(d+1)r^*}$



$\Rightarrow$ $\text{succ}(q)$ must be
in same quadtree cell
(or $\text{pred}(q)$)

$\Rightarrow d(\text{succ}(q), q) \leq \sqrt{d} \cdot 2(d+1)r^*$
(or $\text{pred}(q)$)

$\Rightarrow$ approx factor $\boxed{O(1)}$ ~~6.~~



Rmk - immediate improvable to $O(\log \log U)$
by ID data structures

- can be made dynamic easily!

Q: refine approx factor from $O(1)$ to $1+\epsilon$?

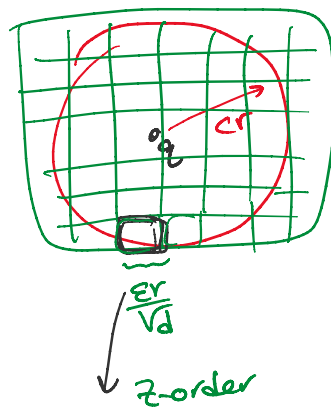
Option 1: let $r =$ factor- c approx.

divide ball (q, cr)
into $O\left(\frac{cr}{\epsilon r/\sqrt{d}}\right)^d$ quadtree cells
of side length $\sim \frac{\epsilon r}{\sqrt{d}}$.
 $= O\left(\frac{1}{\epsilon^d}\right)$

test if each such cell is empty

$\Rightarrow$ query time $\boxed{O\left(\frac{1}{\epsilon^d} \log n\right)}$

or $O\left(\frac{1}{\epsilon^d} \log \log U\right) \dots$



Rmk - best ϵ -dependence

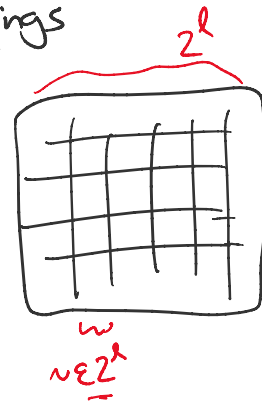
Arya, Mount et al. '94 $O(n)$ space, $O\left(\frac{1}{\epsilon^d} \log n\right)$ time
~ L. nl space $\Omega\left(\frac{1}{\epsilon^{d/2}} \log n\right)$ time

Arya, Mount et al. '94 $O(n)$ space, $O(\frac{1}{\epsilon^d} \log n)$ time
 Clarkson '94 / C. '97 $\tilde{O}(\frac{1}{\epsilon^{d/2}} n)$ space, $O(\frac{1}{\epsilon^{d/2}} \log n)$ time
 ...
 Arya, da Fonseca, Mount '17 / C. '17 $\tilde{O}(\frac{1}{\epsilon^{d/4}} n)$ space, $O(\frac{1}{\epsilon^{d/4}} \log n)$ time

Option 2: C. - HarPeled-Jones '18

idea - keep query alg'm same
but use multiple orderings

increase fan-out to $(\frac{1}{\epsilon})^d$
 (i.e. Compress $\log \frac{1}{\epsilon}$ levels into one)
 reorder children


~~| | | | |
|---|---|----|----|
| 5 | 7 | 13 | 15 |
| 4 | 6 | 12 | 14 |
| 1 | 3 | 9 | 11 |
| 0 | 2 | 8 | 10 |~~

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

0	2	1	3
4	6	5	7
8	10	9	11
12	14	13	15

0	3		

...

$\forall$ children u_i, u_j , $\exists$ ordering st. u_i, u_j adjacent

orderings $O(\frac{1}{\epsilon^d} \log \frac{1}{\epsilon})$
 # levels compressed

many appl's ...

Diameter (farthest pair)

Diameter (farthest pair)

Given n pts in $\mathbb{R}^d$,
find $\Delta^* = \max_{p, q \in P} \|p - q\|$



Known exact alg's: $O(n^2)$ time

$$d=2: O(n \log n)$$

$$d=3: O(n \log n) \text{ not easy!}$$

$$d=4: \tilde{O}(n^{4/3})$$

$$\vdots$$
$$\tilde{O}(n^{2 - \frac{2}{\lfloor d/2 \rfloor + 1}})$$

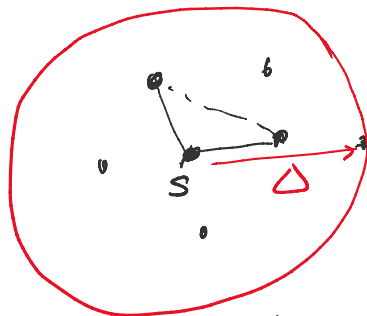
Approx?

Alg'm 0: $s = \text{any pt of } P$
return $\Delta = \text{farthest dist from } s$

Runtime: $O(n)$

Approx factor:

$$P \subseteq \text{ball}(s, \Delta)$$



$$\Rightarrow \Delta \leq \Delta^* \leq 2\Delta \text{ by triangle inequality}$$

$$\Rightarrow \text{factor } 2$$

(works in any metric space)

0
0
0