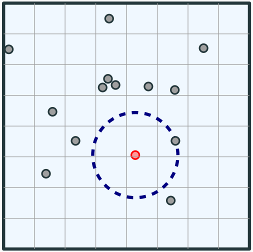
Locality-Sensitive Orderings & Their Applications

Timothy Chan, Sariel Har-Peled, Mitchell Jones

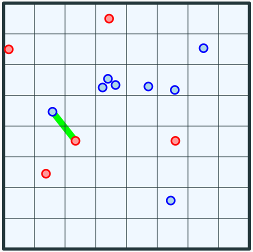
August 2019

University of Illinois at Urbana-Champaign

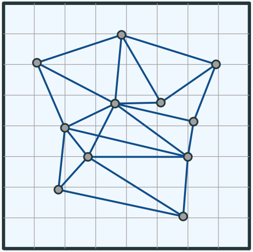
Low dimension proximity problems: $d = O(1)$



Nearest neighbor



Closest pair problems



Spanners/MST

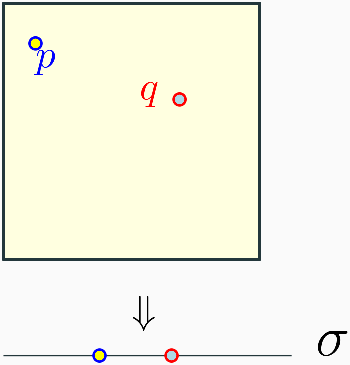
Goal: Dynamic data structures which maintain/return a $(1 + \varepsilon)$-approximation

In this talk

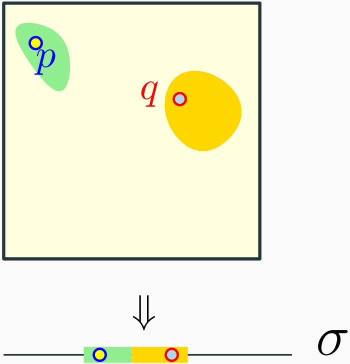
- Quadtrees: Basic data structure in computational geometry
- Many orderings of points in $\mathbb{R}^d$ ($\mathbb{Z}$-order)
- Two new tricks to the mix

  $\Longrightarrow$ Simpler data structures for many proximity problems (plus some new results)

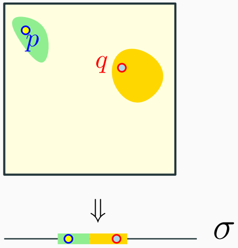
New technique: Locality-sensitive orderings



New technique: Locality-sensitive orderings



New technique: Locality-sensitive orderings



Definition: Locality-Sensitive Orderings

Let $\varepsilon \in (0, 1)$. A collection of orderings Π over $[0, 1)^d$ s.t. for all $p, q \in [0, 1)^d$, exists $\sigma \in \Pi$ where:

$$\forall p \prec_\sigma z \prec_\sigma q : \min(\|z - p\|, \|z - q\|) \leqslant \varepsilon \|p - q\|.$$

New technique: Locality-sensitive orderings

Definition: Locality-Sensitive Orderings

Let $\varepsilon \in (0,1)$. A collection of orderings Π over $[0,1)^d$ s.t. for all $p, q \in [0,1)^d$, exists $\sigma \in \Pi$ where:

$$\forall p \prec_\sigma z \prec_\sigma q : \min(\|z-p\|, \|z-q\|) \leqslant \varepsilon \|p-q\|.$$

Theorem

There are locality-sensitive orderings of size $O\big((1/\varepsilon^d)\log(1/\varepsilon)\big)$.

Main applications

- New: $(1+\varepsilon)$-bichromatic closest pair

Main applications

- New: $(1+\varepsilon)$-bichromatic closest pair
- Simpler: Dynamic $(1+\varepsilon)$-spanners

Main applications

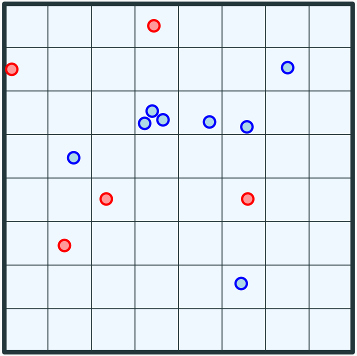
- New: $(1+\varepsilon)$-bichromatic closest pair
- Simpler: Dynamic $(1+\varepsilon)$-spanners
- New: Dynamic k-vertex-fault-tolerant $(1+\varepsilon)$-spanners

Main applications

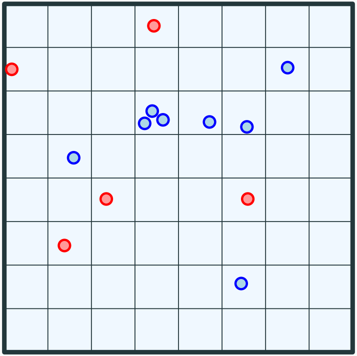
- New: $(1+\varepsilon)$-bichromatic closest pair
- Simpler: Dynamic $(1+\varepsilon)$-spanners
- New: Dynamic k-vertex-fault-tolerant $(1+\varepsilon)$-spanners
- ...

Warmup: Constant factor approximation for bichromatic closest pair

Bichromatic closest pair



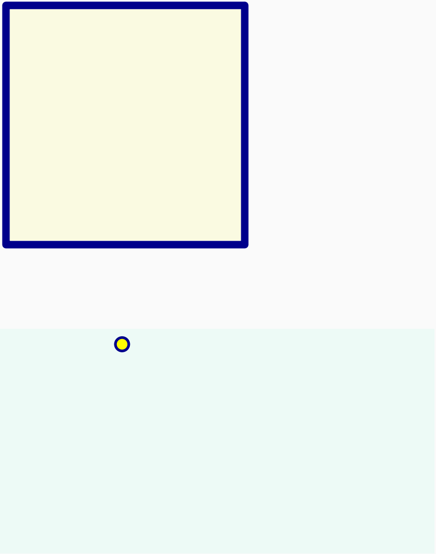
Bichromatic closest pair



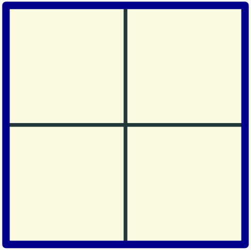
Problem (c-approximation)

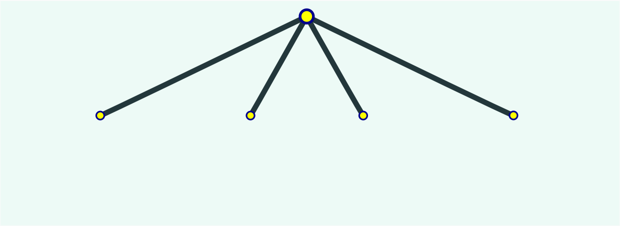
Maintain a pair (r', b') s.t. $\|r' - b'\| \leqslant c \cdot \min_{(r,b)} \|r - b\|$.

Quadtrees

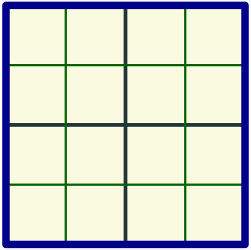


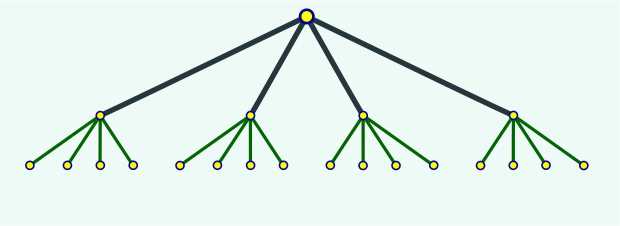
Quadtrees



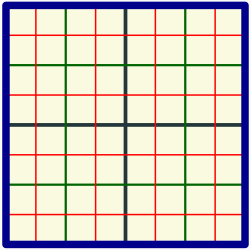


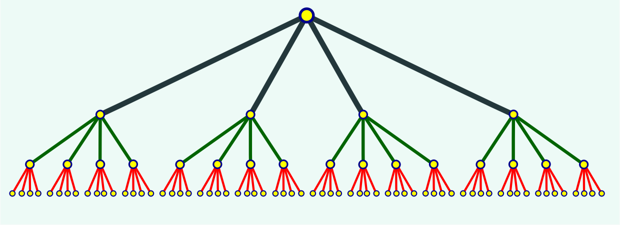
Quadtrees



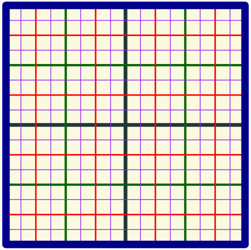


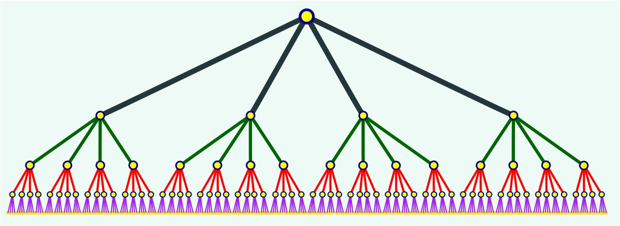
Quadtrees





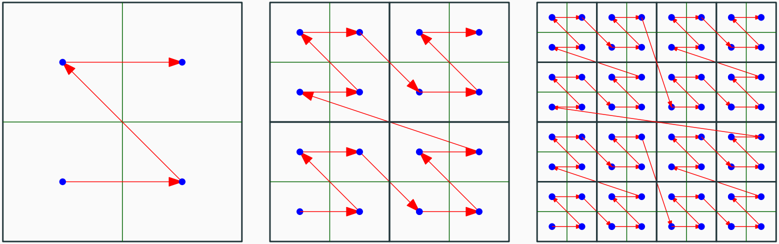
Quadtrees





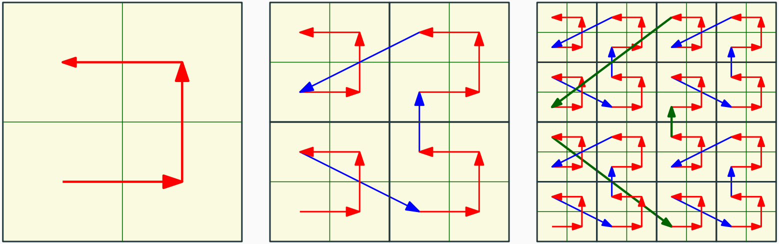
Quadtrees: $\mathbb{Z}$-order

DFS of a quadtree $\implies$ ordering of points ($\mathbb{Z}$-order)

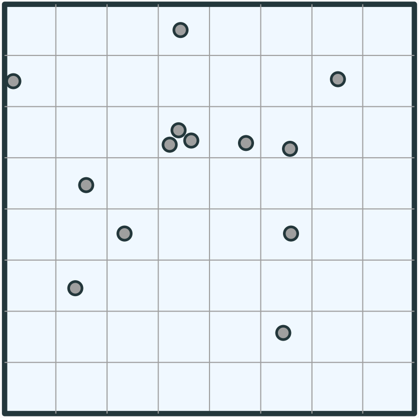


Quadtrees: $\mathcal{Z}$-order

DFS of a quadtree $\implies$ ordering of points ($\supset$-order)

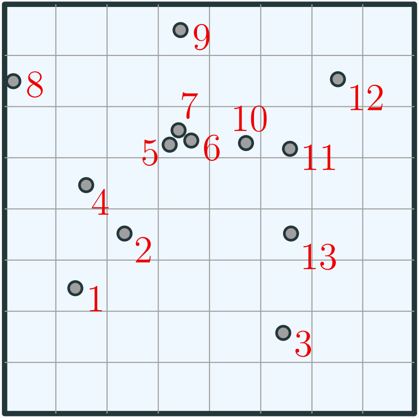


Hope: points close together $\approx$ nearby in ordering



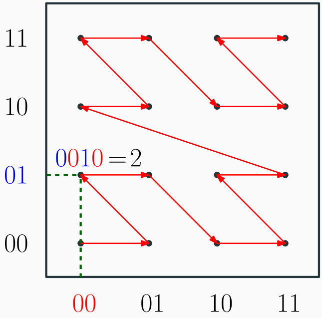
Ordering of points

Hope: points close together $\approx$ nearby in ordering



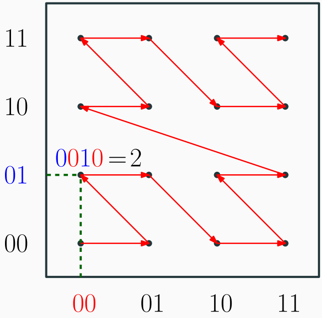
Computing the $\mathbb{Z}$-order

- Let $p = (x, y) \in [2^w] \times [2^w]$
- $x = x_w x_{w-1} \dots x_1$
- $y = y_w y_{w-1} \dots y_1$



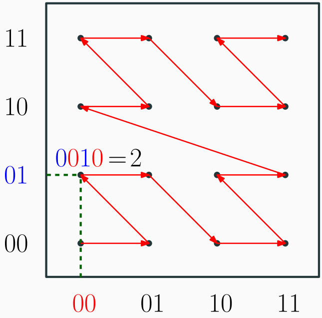
Computing the $\mathcal{Z}$-order

- Let $p = (x, y) \in [2^w] \times [2^w]$
- $x = x_w x_{w-1} \dots x_1$
- $y = y_w y_{w-1} \dots y_1$
- $\text{shuffle}(p) = y_w x_w y_{w-1} x_{w-1} \dots y_1 x_1$
- Position of p in $\mathcal{Z}$-order = $\text{shuffle}(p)$



Computing the $\mathbb{Z}$-order

- Let $p = (x, y) \in [2^w] \times [2^w]$
- $x = x_w x_{w-1} \dots x_1$
- $y = y_w y_{w-1} \dots y_1$
- $\text{shuffle}(p) = y_w x_w y_{w-1} x_{w-1} \dots y_1 x_1$
- Position of p in $\mathbb{Z}$-order = $\text{shuffle}(p)$

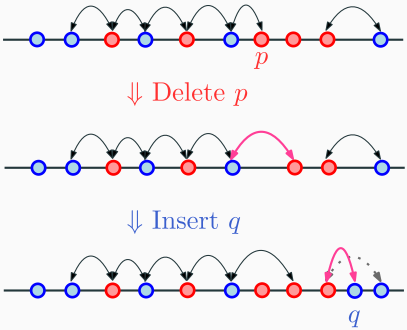


Lemma

$\text{shuffle}(p)$ and $\text{shuffle}(q)$ can be compared with $O(1)$ bitwise-and/xor operations.

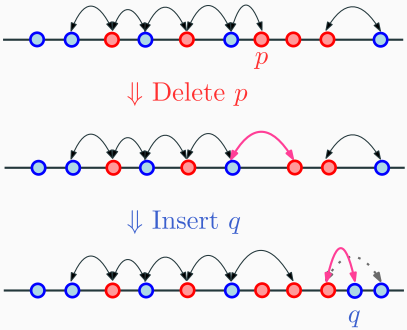
Solving the problem in 1D: A solution?

- Map the point set to 1D



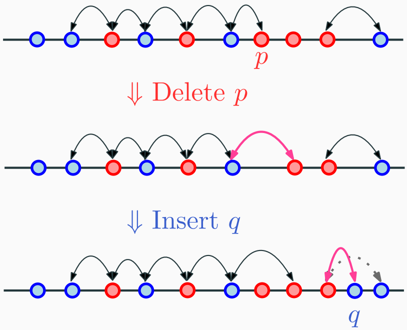
Solving the problem in 1D: A solution?

- Map the point set to 1D
- Maintain sorted order



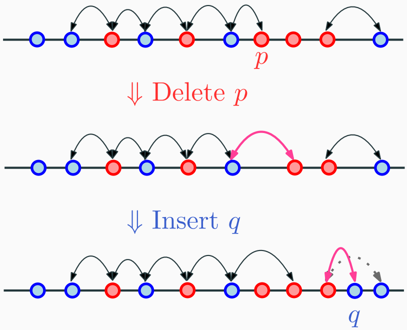
Solving the problem in 1D: A solution?

- Map the point set to 1D
- Maintain sorted order
- Maintain consecutive red/blue pairs with min-heap



Solving the problem in 1D: A solution?

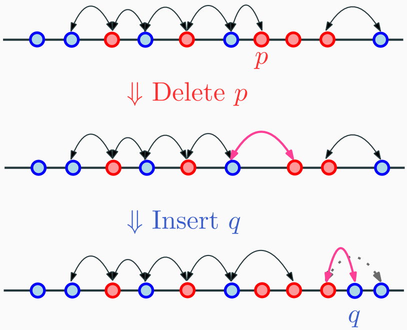
- Map the point set to 1D
- Maintain sorted order
- Maintain consecutive red/blue pairs with min-heap
- Updates change $O(1)$ consecutive pairs



Solving the problem in 1D: A solution?

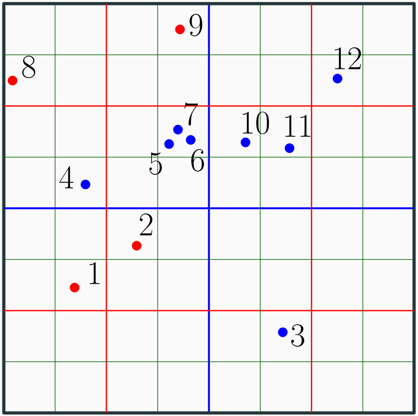
- Map the point set to 1D
- Maintain sorted order
- Maintain consecutive red/blue pairs with min-heap
- Updates change $O(1)$ consecutive pairs

$\Longrightarrow$ Update time $O(\log n)$



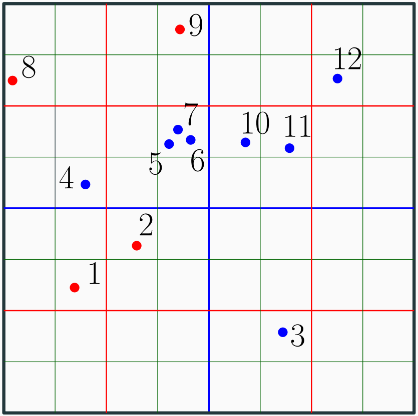
Not quite a solution

- Points nearby in $\mathbb{R}^d \not\Rightarrow$ nearby in $\mathbb{Z}$-order



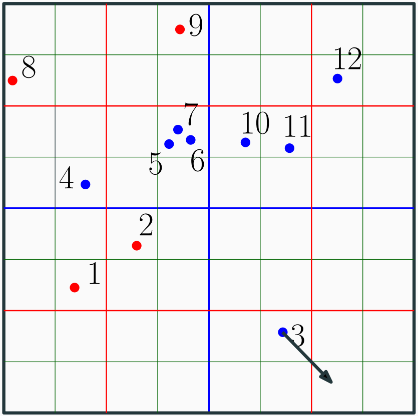
Not quite a solution

- Points nearby in $\mathbb{R}^d \not\Rightarrow$ nearby in $\mathbb{Z}$-order
- Idea: Shift the point set



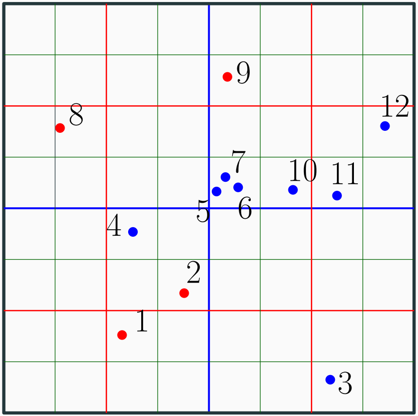
Not quite a solution

- Points nearby in $\mathbb{R}^d \not\Rightarrow$ nearby in $\mathbb{Z}$-order
- Idea: Shift the point set



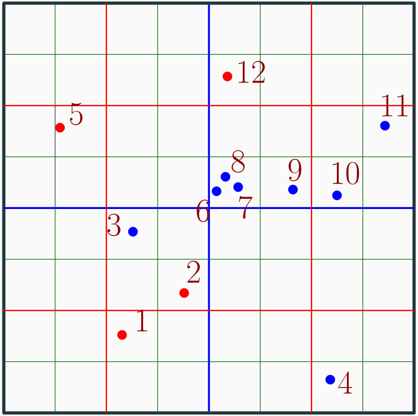
Not quite a solution

- Points nearby in $\mathbb{R}^d$ $\not\Rightarrow$ nearby in $\mathbb{Z}$-order
- Idea: Shift the point set



Not quite a solution

- Points nearby in $\mathbb{R}^d \not\Rightarrow$ nearby in $\mathbb{Z}$-order
- Idea: Shift the point set



Shifting

Lemma [Chan, 1998]

For $i = 0, \ldots, d$, $v_i = (i/(d+1), \ldots, i/(d+1))$.

For any $p, q \in [0,1)^d$, exists $i \in \{0, \ldots, d\}$ and quadtree cell $\square$:

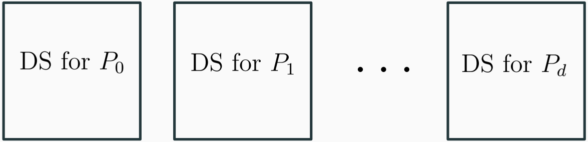
1. $p + v_i, q + v_i \in \square$
2. $(d+1)\|p - q\| < \text{sidelength}(\square) \leqslant 2(d+1)\|p - q\|$.

A correct solution

- Shift point set $d + 1$ times: $P_0, \ldots, P_d$

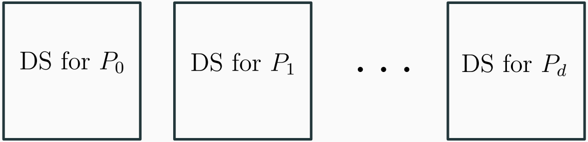
A correct solution

- Shift point set $d+1$ times: $P_0, \ldots, P_d$



A correct solution

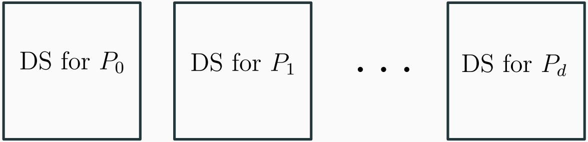
- Shift point set $d + 1$ times: $P_0, \ldots, P_d$



$\implies$ $O_d(\log n)$ update time

A correct solution

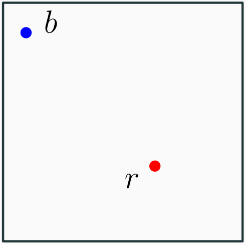
- Shift point set $d + 1$ times: $P_0, \ldots, P_d$



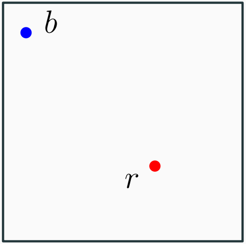
$\implies$ $O_d(\log n)$ update time

- Claim: $O_d(1)$ approximation

Correctness (cont.)

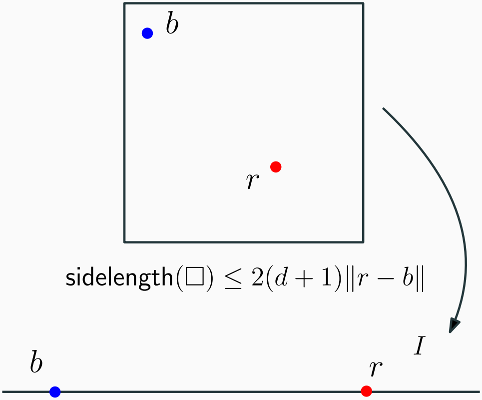


Correctness (cont.)

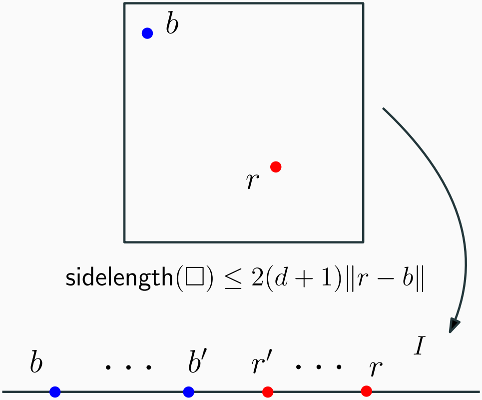


sidelength$(\square) \leq 2(d+1)\|r-b\|$

Correctness (cont.)

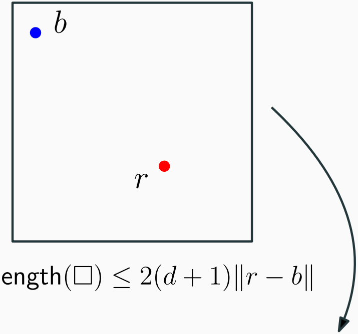


Correctness (cont.)

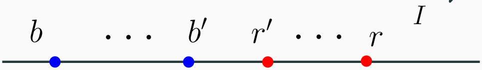


sidelength($\square$) $\leq 2(d+1)\|r-b\|$

Correctness (cont.)



sidelength$(\square) \leq 2(d+1)\|r-b\|$



$\|r'-b'\| \leq$ diam$(\square) \leq \sqrt{d} \cdot$ sidelength$(\square) = O_d(1)\|r-b\|$

The challenge: $(1+\varepsilon)$-approximate bichromatic closest pair

Key idea I: Reducing the approximation factor

- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$

Key idea I: Reducing the approximation factor

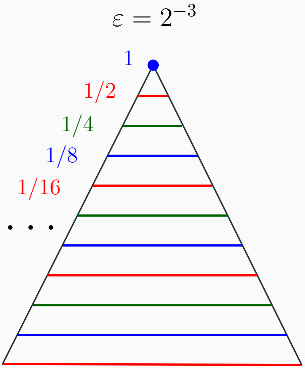
- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$
- Idea: Pack many "ε-quadtrees" into a regular quadtree

Key idea I: Reducing the approximation factor

- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$
- Idea: Pack many "ε-quadtrees" into a regular quadtree
- ε-quadtrees have $1/\varepsilon^d$ children

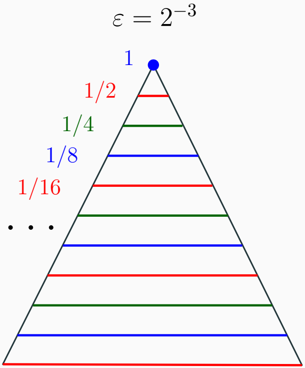
Key idea I: Reducing the approximation factor

- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$
- Idea: Pack many "ε-quadtrees" into a regular quadtree
- ε-quadtrees have $1/\varepsilon^d$ children



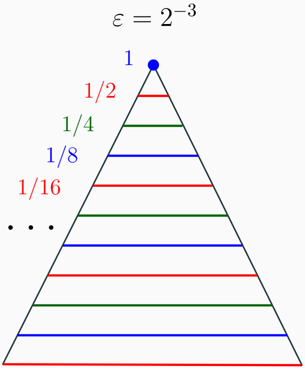
Key idea I: Reducing the approximation factor

- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$
- Idea: Pack many "ε-quadtrees" into a regular quadtree
- ε-quadtrees have $1/\varepsilon^d$ children
- Can partition a regular quadtree into $\lg(1/\varepsilon)$ ε-quadtrees



Key idea I: Reducing the approximation factor

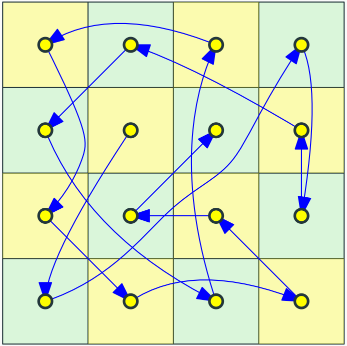
- Assume $\varepsilon = 2^{-E}$ for $E \in \mathbb{N}$
- Idea: Pack many "ε-quadtrees" into a regular quadtree
- ε-quadtrees have $1/\varepsilon^d$ children
- Can partition a regular quadtree into $\lg(1/\varepsilon)$ ε-quadtrees
- Call them $\mathcal{T}_\varepsilon^1, \ldots, \mathcal{T}_\varepsilon^E$



$O(1)$ problems

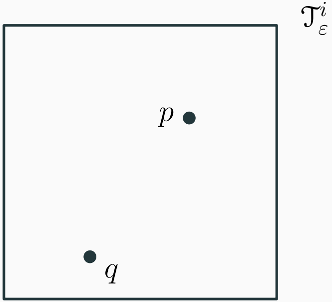
Extend $\mathcal{Z}$-order to ε-quadtrees by ordering $1/\varepsilon^d$ child cells

10	6	9	3
7	1	16	5
11	15	14	4
2	12	8	13



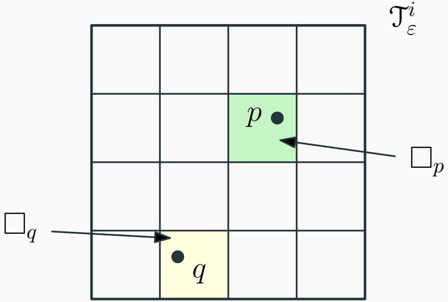
Which order to pick?

$O(1)$ **problems**



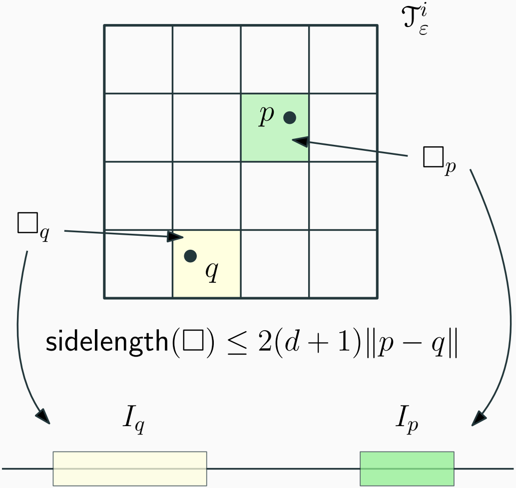
sidelength$(\square) \leq 2(d+1)\|p-q\|$

$O(1)$ problems



sidelength$(\square) \leq 2(d+1)\|p-q\|$

$O(1)$ problems



$\mathcal{T}_\varepsilon^i$
p
q
$\square_p$
$\square_q$
sidelength$(\square) \leq 2(d+1)\|p-q\|$
I_q
I_p

Key idea II: Ordering quadtree cells

Problem

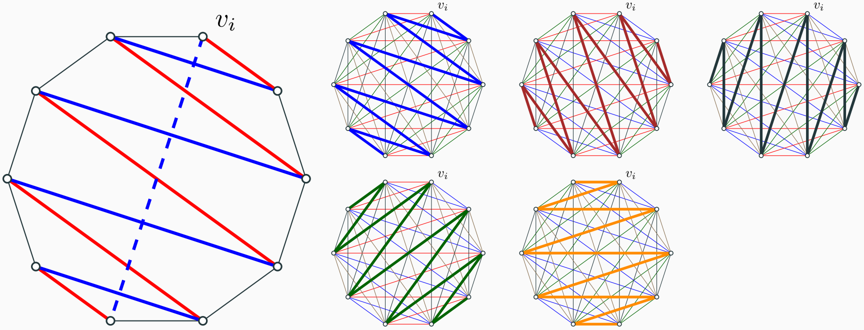
Find a family $\mathfrak{O}$ of orderings of the $1/\varepsilon^d$ cells s.t.:

For any $\square_1, \square_2$, there is an ordering $\sigma \in \mathfrak{O}$ with $\square_1$ adjacent to $\square_2$.

A necessary subproblem

Lemma [Alspach, 2008]

For $[\![n]\!] = \{1, \ldots, n\}$, there are $\lceil n/2 \rceil$ orderings $\mathfrak{O}$ of $[\![n]\!]$ such that for all $i, j \in [\![n]\!]$, $\exists \sigma \in \mathfrak{O}$ where i and j are adjacent in σ.



Ordering quadtree cells

Corollary

There is a set $\mathfrak{O}(\varepsilon)$ of $O(1/\varepsilon^d)$ orderings such that for any $\square_1, \square_2$, there is an order $\sigma \in \mathfrak{O}(\varepsilon)$ where $\square_1$ and $\square_2$ are adjacent.

What we have so far

- $d + 1$ shifted point sets $\equiv d + 1$ quadtrees

What we have so far

- $d+1$ shifted point sets $\equiv d+1$ quadtrees
- Each quadtree has $\lg(1/\varepsilon)$ ε-quadtrees

What we have so far

- $d + 1$ shifted point sets $\equiv d + 1$ quadtrees
- Each quadtree has $\lg(1/\varepsilon)$ ε-quadtrees
- Each ε-quadtree has $O(1/\varepsilon^d)$ orderings

What we have so far

- $d + 1$ shifted point sets $\equiv d + 1$ quadtrees
- Each quadtree has $\lg(1/\varepsilon)$ ε-quadtrees
- Each ε-quadtree has $O(1/\varepsilon^d)$ orderings

$\implies O_d\big((1/\varepsilon^d)\log(1/\varepsilon)\big)$ different orderings of P

What we have so far

- $d+1$ shifted point sets $\equiv d+1$ quadtrees
- Each quadtree has $\lg(1/\varepsilon)$ ε-quadtrees
- Each ε-quadtree has $O(1/\varepsilon^d)$ orderings

  $\implies O_d\big((1/\varepsilon^d)\log(1/\varepsilon)\big)$ different orderings of P
- Π is this family of locality-sensitive orderings

What we have so far

- $d+1$ shifted point sets $\equiv d+1$ quadtrees
- Each quadtree has $\lg(1/\varepsilon)$ ε-quadtrees
- Each ε-quadtree has $O(1/\varepsilon^d)$ orderings

  $\implies O_d\big((1/\varepsilon^d)\log(1/\varepsilon)\big)$ different orderings of P
- Π is this family of locality-sensitive orderings
- For $\sigma \in \Pi$, can decide $p \prec_\sigma q$ with $O(\log(1/\varepsilon))$ bitwise-logical operations.

The solution

- Maintain the 1D data structure for all orderings Π

The solution

- Maintain the 1D data structure for all orderings Π
- $|\Pi| = O((1/\varepsilon^d)\log(1/\varepsilon))$

The solution

- Maintain the 1D data structure for all orderings Π
- $|\Pi| = O((1/\varepsilon^d)\log(1/\varepsilon))$
- Update time: $O(|\Pi| \cdot \log(n) \cdot \log(1/\varepsilon)) = O_d((1/\varepsilon^d)\log(n)\log^2(1/\varepsilon))$

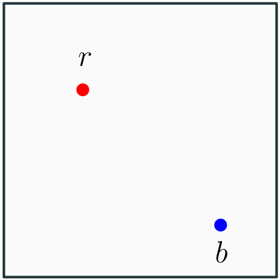
The solution

- Maintain the 1D data structure for all orderings Π
- $|\Pi| = O((1/\varepsilon^d)\log(1/\varepsilon))$
- Update time: $O(|\Pi| \cdot \log(n) \cdot \log(1/\varepsilon)) = O_d((1/\varepsilon^d)\log(n)\log^2(1/\varepsilon))$
- Space: $O(|\Pi| \cdot n) = O_d((n/\varepsilon^d)\log(1/\varepsilon))$

The solution

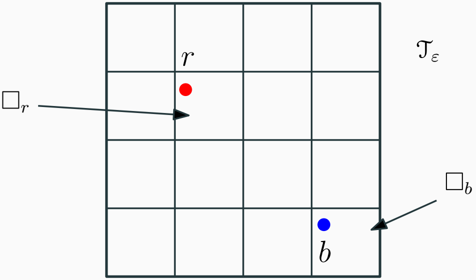
- Maintain the 1D data structure for all orderings Π
- $|\Pi| = O((1/\varepsilon^d)\log(1/\varepsilon))$
- Update time: $O(|\Pi| \cdot \log(n) \cdot \log(1/\varepsilon)) = O_d((1/\varepsilon^d)\log(n)\log^2(1/\varepsilon))$
- Space: $O(|\Pi| \cdot n) = O_d((n/\varepsilon^d)\log(1/\varepsilon))$
- Claim: Maintains r', b' with $\|r' - b'\| \leqslant (1+\varepsilon)\|r - b\|$

Correctness



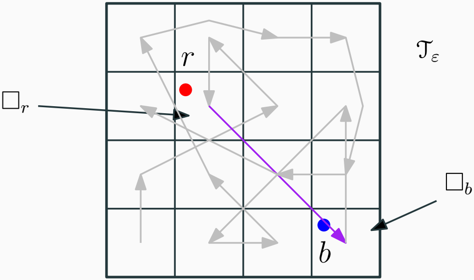
sidelength($\square$) $\leq 2(d+1)\|r-b\|$

Correctness

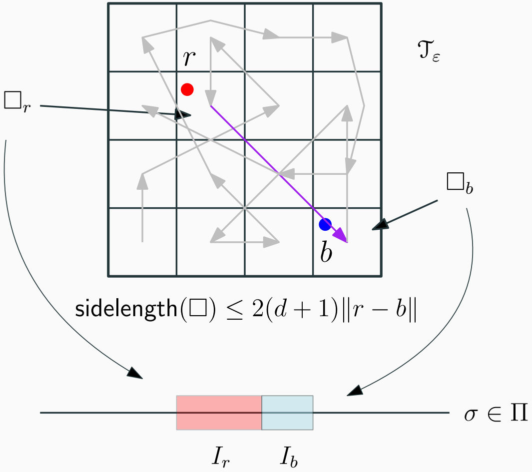


sidelength$(\square) \leq 2(d+1)\|r-b\|$

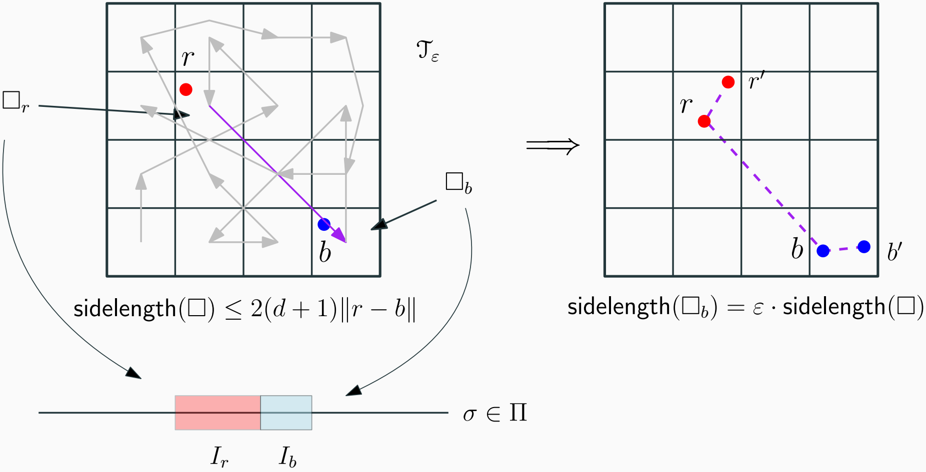
Correctness



sidelength$(\square) \leq 2(d+1)\|r-b\|$


r
b
$\mathcal{T}_\varepsilon$
$\square_r$
$\square_b$
sidelength$(\square) \leq 2(d+1)\|r-b\|$
$\sigma \in \Pi$
I_r
I_b

Correctness



The result

Our result

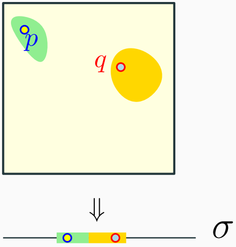
Can maintain the $(1+\varepsilon)$-approximate bichromatic closest pair dynamically with:

1. $O(\log n \log^2(1/\varepsilon)/\varepsilon^d)$ update time
2. $O(n \log(1/\varepsilon)/\varepsilon^d)$ space

Main Theorem

For $\varepsilon \in (0,1)$, there is a set Π of size $O((1/\varepsilon^d)\log(1/\varepsilon))$ s.t. $\forall p, q \in [0,1)^d$, $\exists \sigma \in \Pi$ with:

Points between p and q in σ are distance at most $\varepsilon\|p-q\|$ from p or q.



A simple data structure for dynamic $(1+\varepsilon)$-spanners

Definition

For a set n of P points in $\mathbb{R}^d$ and $t \geqslant 1$, a t-spanner of P is a graph $G = (P, E)$ such that for all $p, q \in P$,

$$\|p - q\| \leqslant \mathrm{dist}_G(p, q) \leqslant t\|p - q\|.$$

Problem

Maintain a $(1 + \varepsilon)$-spanner of P dynamically.

Previous work & result

reference	insertion time	deletion time
[Roditty, 2012]	$O(\log n)$	$O(n^{1/3} \log^{O(1)} n)$
[Gottlieb and Roditty, 2008a]	$O(\log^2 n)$	$O(\log^3 n)$
[Gottlieb and Roditty, 2008b]	$O(\log n)$	$O(\log n)$

Our result

Can dynamically maintain a $(1 + \varepsilon)$-spanner of P with:

1. $O(n \log(1/\varepsilon)/\varepsilon^d)$ edges
2. $O(\log(1/\varepsilon)/\varepsilon^d)$ maximum degree
3. $O(\log n \log^2(1/\varepsilon)/\varepsilon^d)$ update time

Construction

- For each $\sigma \in \Pi$, add edges between consecutive points

Construction

- For each $\sigma \in \Pi$, add edges between consecutive points
- $(n-1)|\Pi| = O_d((n/\varepsilon^d)\log(1/\varepsilon))$ edges

Construction

- For each $\sigma \in \Pi$, add edges between consecutive points
- $(n-1)|\Pi| = O_d((n/\varepsilon^d)\log(1/\varepsilon))$ edges
- Maximum degree $\leqslant 2|\Pi| = O_d((1/\varepsilon^d)\log(1/\varepsilon))$

Construction

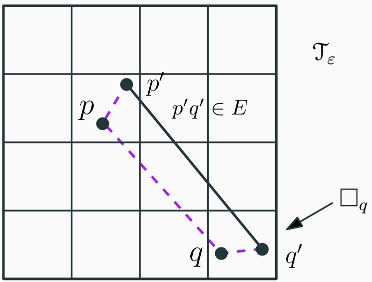
- For each $\sigma \in \Pi$, add edges between consecutive points
- $(n-1)|\Pi| = O_d((n/\varepsilon^d)\log(1/\varepsilon))$ edges
- Maximum degree $\leqslant 2|\Pi| = O_d((1/\varepsilon^d)\log(1/\varepsilon))$
- Update time $O_d((1/\varepsilon^d)\log(n)\log^2(1/\varepsilon))$

Construction

- For each $\sigma \in \Pi$, add edges between consecutive points
- $(n-1)|\Pi| = O_d((n/\varepsilon^d)\log(1/\varepsilon))$ edges
- Maximum degree $\leqslant 2|\Pi| = O_d((1/\varepsilon^d)\log(1/\varepsilon))$
- Update time $O_d((1/\varepsilon^d)\log(n)\log^2(1/\varepsilon))$
- Claim: G is a $(1+\varepsilon)$-spanner

Proof idea

- Proof by induction on length of pairs:
  $\mathrm{dist}_G(p, q) \leqslant (1 + \varepsilon)\|p - q\|$

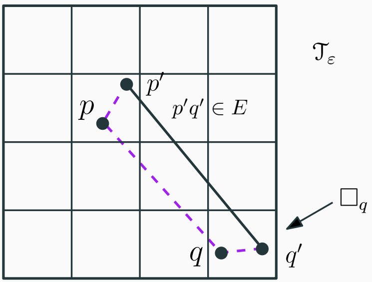


sidelength($\square$) $\leq 2(d+1)\|p-q\|$

sidelength($\square_q$) $= \varepsilon \cdot$ sidelength($\square$)

Proof idea

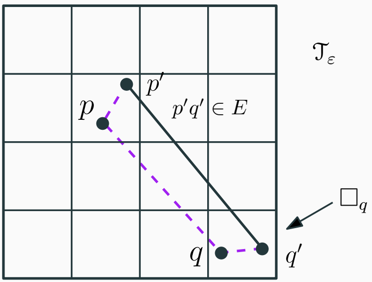
- Proof by induction on length of pairs: $\text{dist}_G(p, q) \leqslant (1 + \varepsilon)\|p - q\|$
- G is a $(1 + c_d\varepsilon)$-spanner for const. c_d



$$\text{sidelength}(\square) \leq 2(d+1)\|p-q\|$$
$$\text{sidelength}(\square_q) = \varepsilon \cdot \text{sidelength}(\square)$$

Proof idea

- Proof by induction on length of pairs: $\text{dist}_G(p, q) \leqslant (1 + \varepsilon)\|p - q\|$
- G is a $(1 + c_d\varepsilon)$-spanner for const. c_d
- Readjust ε by c_d



$\text{sidelength}(\square) \leq 2(d+1)\|p - q\|$

$\text{sidelength}(\square_q) = \varepsilon \cdot \text{sidelength}(\square)$

Static & dynamic vertex-fault-tolerant spanners

Fault-tolerant spanners

Definition

For a set of n points P in $\mathbb{R}^d$ and $t \geqslant 1$, a k-vertex-fault-tolerant t-spanner of P is a graph $G = (P, E)$ such that

1. G is a t-spanner, and
2. For any $P' \subseteq P$, $|P'| \leqslant k$, $G \setminus P'$ is a t-spanner for $P \setminus P'$.

Problem

For a static point set P, efficiently construct a “small” k-VFT $(1 + \varepsilon)$-spanner.

Previous work & result

reference	# edges	degree	running time
[Levcopoulos et al., 1998]	$2^{O(k)}n$	$2^{O(k)}$	$O(n\log n + 2^{O(k)}n)$
	$O(k^2 n)$	unbounded	$O(n\log n + k^2 n)$
	$O(kn\log n)$	unbounded	$O(kn\log n)$
[Lukovszki, 1999]	$O(kn)$	$O(k^2)$	$O(n\log^{d-1} n + kn\log\log n)$
[Czumaj and Zhao, 2004]	$O(kn)$	$O(k)$	$O(kn\log^d n + k^2 n\log k)$
[Chan et al., 2015]	$O(k^2 n)$	$O(k^2)$	$O(n\log n + k^2 n)$
[Kapoor and Li, 2013] & [Solomon, 2014]	$O(kn)$	$O(k)$	$O(n\log n + kn)$

Our result

A k-VFT $(1+\varepsilon)$-spanner of P with

1. $O(kn\log(1/\varepsilon)/\varepsilon^d)$ edges
2. $O(k\log(1/\varepsilon)/\varepsilon^d)$ maximum degree
3. $O\big((n\log n\log(1/\varepsilon) + kn)\log(1/\varepsilon)/\varepsilon^d\big)$ construction time

Construction

- For each $\sigma \in \Pi$ and each $p \in P$, connect p to its $k+1$ predecessors and successors in σ

Construction

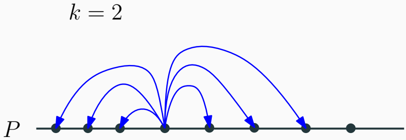
- For each $\sigma \in \Pi$ and each $p \in P$, connect p to its $k+1$ predecessors and successors in σ
- $O(kn|\Pi|) = O_d((kn/\varepsilon^d)\log(1/\varepsilon))$ edges

Construction

- For each $\sigma \in \Pi$ and each $p \in P$, connect p to its $k + 1$ predecessors and successors in σ
- $O(kn|\Pi|) = O_d((kn/\varepsilon^d) \log(1/\varepsilon))$ edges
- Maximum degree = $O(k|\Pi|) = O_d((k/\varepsilon^d) \log(1/\varepsilon))$

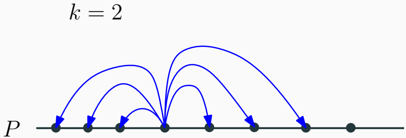
Sketch proof

- G is a $(1+\varepsilon)$-spanner



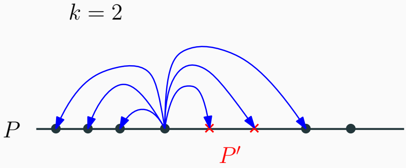
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$



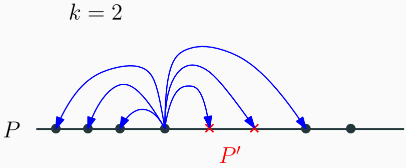
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$



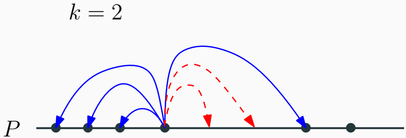
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$
- Let $\sigma \in \Pi$ with P' removed



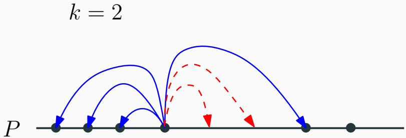
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$
- Let $\sigma \in \Pi$ with P' removed



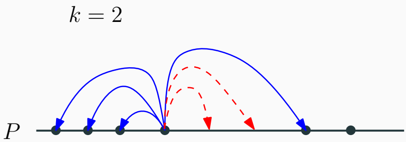
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$
- Let $\sigma \in \Pi$ with P' removed
- Consecutive points in $P \setminus P'$ remain in $G \setminus P'$ (by construction)



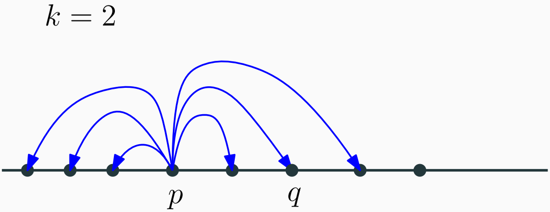
Sketch proof

- G is a $(1+\varepsilon)$-spanner
- Consider $P' \subseteq P$, $|P'| \leqslant k$
- Let $\sigma \in \Pi$ with P' removed
- Consecutive points in $P \setminus P'$ remain in $G \setminus P'$ (by construction)
  $\implies G \setminus P'$ is a $(1+\varepsilon)$-spanner for $P \setminus P'$



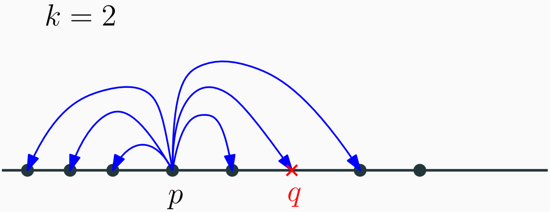
Update time

Any update changes $O(k)$ edges in G



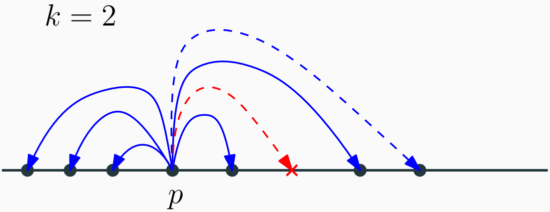
Update time

Any update changes $O(k)$ edges in G



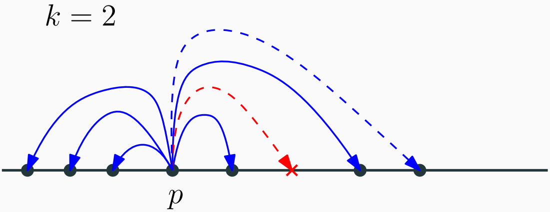
Update time

Any update changes $O(k)$ edges in G



Update time

Any update changes $O(k)$ edges in G



Update time $O_d((\log n \log(1/\varepsilon) + k)|\Pi|)$

Result

Our result

A k-VFT $(1+\varepsilon)$-spanner of P with

1. $O(kn \log(1/\varepsilon)/\varepsilon^d)$ edges
2. $O(k \log(1/\varepsilon)/\varepsilon^d)$ maximum degree
3. $O\big((n \log n \log(1/\varepsilon) + kn) \log(1/\varepsilon)/\varepsilon^d\big)$ construction time

New: Can also maintain dynamically with update time

$$O\Big((\log n \log(1/\varepsilon) + k) \log(1/\varepsilon)/\varepsilon^d\Big).$$

Conclusion

Main Theorem

Main Theorem

For $\varepsilon \in (0, 1)$, there is a set Π of size $O((1/\varepsilon^d) \log(1/\varepsilon))$ s.t. $\forall p, q \in [0, 1)^d$, $\exists \sigma \in \Pi$ with:

Points between p and q in σ are distance at most $\varepsilon\|p - q\|$ from p or q.

Remarks

- Extends to $\| \cdot \|_p$ norms

Main Theorem

Main Theorem

For $\varepsilon \in (0,1)$, there is a set Π of size $O((1/\varepsilon^d)\log(1/\varepsilon))$ s.t.
$\forall p, q \in [0,1)^d$, $\exists \sigma \in \Pi$ with:

Points between p and q in σ are distance at most $\varepsilon\|p-q\|$ from p or q.

Remarks

- Extends to $\|\cdot\|_p$ norms
- "Replacement" for well-separated pair decomposition

Main Theorem

For $\varepsilon \in (0,1)$, there is a set Π of size $O((1/\varepsilon^d)\log(1/\varepsilon))$ s.t. $\forall p, q \in [0,1)^d$, $\exists \sigma \in \Pi$ with:

Points between p and q in σ are distance at most $\varepsilon\|p-q\|$ from p or q.

Remarks

- Extends to $\|\cdot\|_p$ norms
- “Replacement” for well-separated pair decomposition
- $\approx$ locality-sensitive hashing (smaller family of orders, weaker guarantees)

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\rightarrow O(\log n)$

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\rightarrow O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\rightarrow O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]
3. Static vertex-fault-tolerant spanners: Simpler in Euclidean setting, matches best known [Kapoor and Li, 2013] and [Solomon, 2014]

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\rightarrow O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]
3. Static vertex-fault-tolerant spanners: Simpler in Euclidean setting, matches best known [Kapoor and Li, 2013] and [Solomon, 2014]
4. Dynamic vertex-fault-tolerant spanners: New

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\to O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]
3. Static vertex-fault-tolerant spanners: Simpler in Euclidean setting, matches best known [Kapoor and Li, 2013] and [Solomon, 2014]
4. Dynamic vertex-fault-tolerant spanners: New
5. Approximate nearest neighbor: Not new

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\rightarrow O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]
3. Static vertex-fault-tolerant spanners: Simpler in Euclidean setting, matches best known [Kapoor and Li, 2013] and [Solomon, 2014]
4. Dynamic vertex-fault-tolerant spanners: New
5. Approximate nearest neighbor: Not new
6. Dynamic approximate MST: Follows by dynamic spanners

Applications

1. Approximate bichromatic closest pair: Improved update time $\approx O(\log^3 n)$ [Eppstein, 1995] $\to O(\log n)$
2. Dynamic spanners: Simpler in Euclidean setting, matches best known [Gottlieb and Roditty, 2008b]
3. Static vertex-fault-tolerant spanners: Simpler in Euclidean setting, matches best known [Kapoor and Li, 2013] and [Solomon, 2014]
4. Dynamic vertex-fault-tolerant spanners: New
5. Approximate nearest neighbor: Not new
6. Dynamic approximate MST: Follows by dynamic spanners
7. Static robust $(1+\varepsilon)$-spanners: See [Buchin et al., 2018]

References i

Timothy M. Chan. *Approximate nearest neighbor queries revisited.* *Discrete Comput. Geom.*, 20(3): 359–373, 1998.

Brian Alspach. *The wonderful Walecki construction.* *Bull. Inst. Combin. Appl.*, 52: 7–20, 2008. ISSN: 1183-1278.

Liam Roditty. *Fully dynamic geometric spanners.* *Algorithmica*, 62(3-4): 1073–1087, 2012.

Lee-Ad Gottlieb and Liam Roditty. *Improved algorithms for fully dynamic geometric spanners and geometric routing.* *Proc. 19th ACM-SIAM Sympos. Discrete Alg.* (SODA), 591–600, 2008.

Lee-Ad Gottlieb and Liam Roditty. *An optimal dynamic spanner for doubling metric spaces.* *Proc. 16th Annu. Euro. Sympos. Alg.* (ESA), 478–489, 2008.

Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. *Efficient algorithms for constructing fault-tolerant geometric spanners*. *Proc. 30th ACM Sympos. Theory Comput.* (STOC), 186–195, 1998.

Tamás Lukovszki. *New results of fault tolerant geometric spanners*. *Proc. 6th Workshop Alg. Data Struct.* (WADS), vol. 1663. 193–204, 1999.

Artur Czumaj and Hairong Zhao. *Fault-tolerant geometric spanners*. *Discrete Comput. Geom.*, 32(2): 207–230, 2004.

T.-H. Hubert Chan, Mingfei Li, Li Ning, and Shay Solomon. *New doubling spanners: Better and simpler*. *SIAM J. Comput.*, 44(1): 37–53, 2015.

Sanjiv Kapoor and Xiang-Yang Li. *Efficient construction of spanners in d-dimensions*. *CoRR*, abs/1303.7217, 2013.

References iii

- Shay Solomon. *From hierarchical partitions to hierarchical covers: Optimal fault-tolerant spanners for doubling metrics.* *Proc. 46th ACM Sympos. Theory Comput.* (STOC), 363–372, 2014.
- David Eppstein. *Dynamic Euclidean minimum spanning trees and extrema of binary functions.* *Discrete Comput. Geom.*, 13: 111–122, 1995.
- Kevin Buchin, Sariel Har-Peled, and Dániel Oláh. *A spanner for the day after.* *CoRR*, abs/1811.06898, 2018. arXiv: `1811.06898`.