Applications of Well-Separated Pairs

Faranak Tohidi

Department of Computer Science
Yazd University

January 18, 2016

Outline

Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

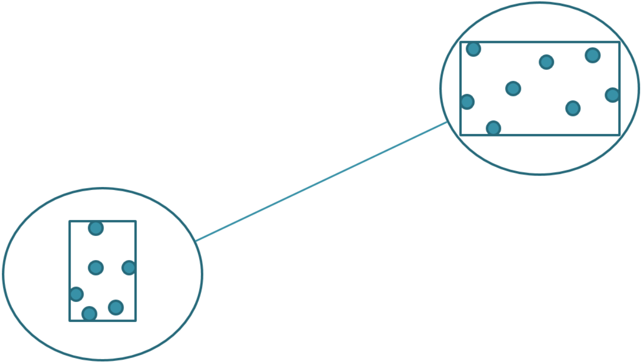
Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

Well-Separated Pairs

- Let $s > 0$ be a real number and A, B are two finite sets of points in R^d.
- A and B are well separated with respect to s if there are 2 disjoint d-dimensional balls Ca and Cb such that:
- Ca and Cb have the same radius
- Ca contains the axes parallel bounding box $R(A)$ of A
- Cb contains the axes parallel bounding box $R(B)$ of B
- The distance between Ca and Cb is greater than or equal to s times the radius of Ca

Well-Separated Pairs



WSPD

- Let S be a set of n points in R^d, and let $s > 0$ be a real number. A WSPD for S in respect to s is a sequence $\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$ of pairs of nonempty subsets of S, for some integer m such that:
- For each $1 \leq i \leq m$, A_i and B_i are well separated with respect to s
- For any two distinct points p and q, there is exactly one index $1 \leq i \leq m$ such that $p \in A_i$ and $q \in B_i$ or the opposite
- The integer m is called the size of the WSPD

WSPD

- Let S be a set of n points in R^d, and let $s > 0$ be a real number. A WSPD for S in respect to s is a sequence $\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$ of pairs of nonempty subsets of S, for some integer m such that:
- For each $1 \leq i \leq m$, A_i and B_i are well separated with respect to s
- For any two distinct points p and q, there is exactly one index $1 \leq i \leq m$ such that $p \in A_i$ and $q \in B_i$ or the opposite
- The integer m is called the size of the WSPD

WSPD

- Let S be a set of n points in R^d, and let $s > 0$ be a real number. A WSPD for S in respect to s is a sequence $\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$ of pairs of nonempty subsets of S, for some integer m such that:
- For each $1 \leq i \leq m$, A_i and B_i are well separated with respect to s
- For any two distinct points p and q, there is exactly one index $1 \leq i \leq m$ such that $p \in A_i$ and $q \in B_i$ or the opposite
- The integer m is called the size of the WSPD

WSPD

- Let S be a set of n points in R^d, and let $s > 0$ be a real number. A WSPD for S in respect to s is a sequence $\{A_1, B_1\}, \{A_2, B_2\}, \dots, \{A_m, B_m\}$ of pairs of nonempty subsets of S, for some integer m such that:
- For each $1 \leq i \leq m$, A_i and B_i are well separated with respect to s
- For any two distinct points p and q, there is exactly one index $1 \leq i \leq m$ such that $p \in A_i$ and $q \in B_i$ or the opposite
- The integer m is called the size of the WSPD

Split Tree

1. $R(S)$ is the bounding box of the set S
2. The split tree for S is a binary tree containing the points of S in its leaves
3. If $S = 1$, the node consists of the single point
4. Else, the node consists of S, and its 2 sons consist of 2 equal halves $S1$ $S2$.
5. The split tree is used to find WSPD

Split Tree

1. $R(S)$ is the bounding box of the set S
2. The split tree for S is a binary tree containing the points of S in its leaves
3. If $S = 1$, the node consists of the single point
4. Else, the node consists of S, and its 2 sons consist of 2 equal halves $S1$ $S2$.
5. The split tree is used to find WSPD

Split Tree

1. $R(S)$ is the bounding box of the set S
2. The split tree for S is a binary tree containing the points of S in its leaves
3. If $S = 1$, the node consists of the single point
4. Else, the node consists of S, and its 2 sons consist of 2 equal halves $S1$ $S2$.
5. The split tree is used to find WSPD

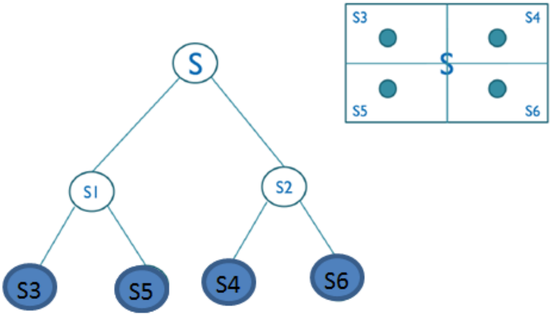
Split Tree

1. $R(S)$ is the bounding box of the set S
2. The split tree for S is a binary tree containing the points of S in its leaves
3. If $S = 1$, the node consists of the single point
4. Else, the node consists of S, and its 2 sons consist of 2 equal halves $S1$ $S2$.
5. The split tree is used to find WSPD

Split Tree

1. $R(S)$ is the bounding box of the set S
2. The split tree for S is a binary tree containing the points of S in its leaves
3. If $S = 1$, the node consists of the single point
4. Else, the node consists of S, and its 2 sons consist of 2 equal halves $S1$ $S2$.
5. The split tree is used to find WSPD

Split Tree



Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

Main Idea of spanner construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD from T
4. Choose the pairs representatives wisely
5. Connect the representatives
6. We get a t-spanner with special properties

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \dfrac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \dfrac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \dfrac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \dfrac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \frac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. Let S be a set of n points in R^d
2. Compute a split tree T for S
3. Construct a WSPD with $s = \frac{4(\sqrt{t}+1)}{(\sqrt{t}-1)}$
4. size of WSPD: $m = O(s^d n)$
5. Time complexity $O(nlogn + s^d n)$
6. As (section 9.2), $G = (S, E)$, where $E := \{\{a_i, b_i\} : 1 \leq i \leq m\}$, is a $\sqrt{t}$-spanner for S

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

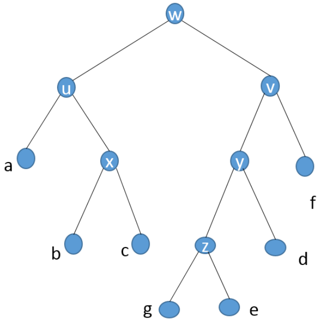
1. For each leaf u in T, let $r(u)$ be the point of S stored in it.
2. For each internal node u of T, let $r(u)$ be the point of S that is stored in the rightmost leaf of the left sub-tree of u.
3. Since every internal node of T has 2 children, $r(u)$ is distinct for every internal node.
4. in $O(n)$ total time can compute $r(u)$ for each node of T.
5. Let i be $1 \leq i \leq m$, the pair A_i, B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
6. A_i,B_i are the set of points stored in the sub-trees of u_i and v_i.
7. We take $r(u_i)A_i$ to be its representative. We do the same for $r(v_i)B_i$.

The first construction

- In $O(n)$ total time, we can compute, for each node u of T, the point $r(u)$, and store t with u.

$$r(a) = a$$
$$r(u) = a$$
$$r(v) = d$$

Inorder traversal of T :aubxcwgzeydvf



The first construction

Lemma 10.1.1

Let p be a point of S. There are at most 2 nodes u in T for which $r(u) = p$

Proof.

Follows from the way we selected the representatives. □

The first construction

Lemma 10.1.1

Let p be a point of S. There are at most 2 nodes u in T for which $r(u) = p$

Proof.

Follows from the way we selected the representatives. □

The first construction

- For each i,$1 \leq i \leq m$, we direct the pair $\{A_i, B_i\}$
- as (lemma 9.4.4). We denote the directed graph as G'

Lemma 10.1.2

The out degree of ofeach vertex in G' is less than or equal to $2((2s+4)\sqrt{d}+4)^d$

Proof.

- Let p be an arbitrary vertex in G'. We have seen that there are at most 2 nodes u in T such that $r(u) = p$
- As shown (lemma 9.4.5 the packing lemma), for each node u in T, there are at most $((2s+4)\sqrt{d}+4)^d$ nodes v in T such that (S_u, S_v) is a pair in WSPD.

□

The first construction

- For each i,$1 \leq i \leq m$, we direct the pair $\{A_i, B_i\}$
- as (lemma 9.4.4). We denote the directed graph as G'

Lemma 10.1.2

The out degree of ofeach vertex in G' is less than or equal to $2((2s+4)\sqrt{d}+4)^d$

Proof.

- Let p be an arbitrary vertex in G'. We have seen that there are at most 2 nodes u in T such that $r(u) = p$
- As shown (lemma 9.4.5 the packing lemma), for each node u in T, there are at most $((2s+4)\sqrt{d}+4)^d$ nodes v in T such that (S_u, S_v) is a pair in WSPD.

□

The first construction

- For each i,$1 \leq i \leq m$, we direct the pair $\{A_i, B_i\}$
- as (lemma 9.4.4). We denote the directed graph as G'

Lemma 10.1.2

The out degree of ofeach vertex in G' is less than or equal to $2((2s+4)\sqrt{d}+4)^d$

Proof.

- Let p be an arbitrary vertex in G'. We have seen that there are at most 2 nodes u in T such that $r(u) = p$
- As shown (lemma 9.4.5 the packing lemma), for each node u in T, there are at most $((2s+4)\sqrt{d}+4)^d$ nodes v in T such that (S_u, S_v) is a pair in WSPD.

□

The first construction

- For each i,$1 \leq i \leq m$, we direct the pair $\{A_i, B_i\}$
- as (lemma 9.4.4). We denote the directed graph as G'

Lemma 10.1.2

The out degree of ofeach vertex in G' is less than or equal to $2((2s+4)\sqrt{d}+4)^d$

Proof.

- Let p be an arbitrary vertex in G'. We have seen that there are at most 2 nodes u in T such that $r(u) = p$
- As shown (lemma 9.4.5 the packing lemma), for each node u in T, there are at most $((2s+4)\sqrt{d}+4)^d$ nodes v in T such that (S_u, S_v) is a pair in WSPD.

□

The first construction

- For each i,$1 \leq i \leq m$, we direct the pair $\{A_i, B_i\}$
- as (lemma 9.4.4). We denote the directed graph as G'

Lemma 10.1.2

The out degree of ofeach vertex in G' is less than or equal to $2((2s+4)\sqrt{d}+4)^d$

Proof.

- Let p be an arbitrary vertex in G'. We have seen that there are at most 2 nodes u in T such that $r(u) = p$
- As shown (lemma 9.4.5 the packing lemma), for each node u in T, there are at most $((2s+4)\sqrt{d}+4)^d$ nodes v in T such that (S_u, S_v) is a pair in WSPD.

□

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The first construction

Lemma 9.4.5

Let a be any node of the split tree T .There are at most $((2s+4)\sqrt{d}+4)^d$ nodes b in T such that (S_a, S_b) is a directed pair in the WSPD computed by algorithm ComputeWSPD(T, s).

Time Complexity

- WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6)
- Representative selection: $O(n)$
- Undirected t-spanner of bounded degree transformation: $O(nlogn)$
- All in all : $O(nlogn)$

The second construction

Lemma 10.1.4

Let θ, w, and t be real numbers such that $0 < \theta < \pi/4$, $0 \leq w < (\cos\theta - \sin\theta)/2$, and $t \geq 1/(\cos\theta - \sin\theta - 2w)$. Let S be a set of n points in the R^d, and let $G = (S, E)$ be a directed graph, such that the following holds: For any two distinct points p and q of S, there is an edge $(r, s) \in E$, such that

1. angle $(pq, xy) \leq \theta$,
2. $|xy| \leq |pq| / \cos\theta$, and
3. $|px| \leq w|xy|$

Then, the graph G is a t-spanner for S.

The second construction

Let S be a set of n points in R^d , and let $t > 1$ be a real number

- $0 < \theta < \pi/4$ and $\cos\theta - \sin\theta > 1/t$
- $s := max(\frac{4\cos\theta}{1-\cos\theta}, \frac{4}{\sin(\theta/2)}, \frac{4}{\cos\theta - \sin\theta - 1/t})$
- $w = 2/s$

Lemma 10.1.5

1. $1 + 4/s \leq 1/\cos\theta$
2. $4/s \leq sin(\theta/2)$
3. $t \geq 1/(\cos\theta - \sin\theta - 2w)$
4. $0 < w < (\cos\theta - \sin\theta)/2$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \leq i \leq m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \leq i \leq m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \leq i \leq m\} \cup \{(b_i, a_i) : 1 \leq i \leq m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second construction

- compute the split tree T together with the corresponding WSPD
- For each node u of T , let S_u be the set of all points of S that are stored in the subtree of u.
- For each i with $1 \le i \le m$, let u_i and v_i be the nodes of T such that $S_{u\,i} = A_i$ and $S_{v\,i} = B_i$.
- choose the representatives $a_i \in A_i$, $b_i \in B_i$,$1 \le i \le m$ by picking the rightmost leaf of the left subtree
- Let $G_0 = (S, E_0)$ be the directed graph with edge set
- $E_0 = \{(a_i, b_i) : 1 \le i \le m\} \cup \{(b_i, a_i) : 1 \le i \le m\}$
- We number the edges of E_0 arbitrarily, from one to $|E_0|$

The second conestruction

- Let $k = k_{d,\theta/2}$
- Let C_k be the $\frac{\theta}{2}$-frame consisting of $k = O(1/\theta^{d-1})$ cones.
- C_k is a collection of k simplicial cones of angular diameter $\frac{\theta}{2}$, having their apex at the origin, and that cover R^d. For each cone $C \in C_k$, let
- $E_0(C) = \{(p, q) \in E_0 : q - p \in C\}$

The second conestruction

- Let $k = k_{d,\theta/2}$
- Let C_k be the $\frac{\theta}{2}$-frame consisting of $k = O(1/\theta^{d-1})$ cones.
- C_k is a collection of k simplicial cones of angular diameter $\frac{\theta}{2}$, having their apex at the origin, and that cover R^d. For each cone $C \in C_k$, let
- $E_0(C) = \{(p, q) \in E_0 : q - p \in C\}$

The second conestruction

- Let $k = k_{d,\theta/2}$
- Let C_k be the $\frac{\theta}{2}$-frame consisting of $k = O(1/\theta^{d-1})$ cones.
- C_k is a collection of k simplicial cones of angular diameter $\frac{\theta}{2}$, having their apex at the origin, and that cover R^d. For each cone $C \in C_k$, let
- $E_0(C) = \{(p, q) \in E_0 : q - p \in C\}$

The second conestruction

- Let $k = k_{d,\theta/2}$
- Let C_k be the $\frac{\theta}{2}$-frame consisting of $k = O(1/\theta^{d-1})$ cones.
- C_k is a collection of k simplicial cones of angular diameter $\frac{\theta}{2}$, having their apex at the origin, and that cover R^d. For each cone $C \in C_k$, let
- $E_0(C) = \{(p,q) \in E_0 : q - p \in C\}$

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second conestruction

For a fixed cone $C \in C_k$ and a fixed point $p \in S$ Consider all nodes u of the split tree T such that:

1. u is on the path from the root to the leaf storing p
2. there is a node v in T , such that
   - $\{S_u, S_v\}$ is a pair in our WSPD, and
   - if x_u and y_v are the representatives of S_u and S_v, respectively, then edge (x_u, y_v) is contained in $E_0(C)$.
3. among all edges (x_u, y_v) obtained in this way, we mark the shortest one. In case of ties, we mark the shortest edge whose index is minimum.
4. Let E be the set of all marked edges. the directed graph $G = (S, E)$ is a t-spanner for S

The second construction

Lemma 10.1.6

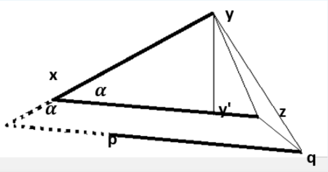
Let A and B be two finite sets of points in R^d that are well-separated with respect to a separation ratio $S > 0$. Let p and x be two points in A , and let q and y be two points in B. Then,

$$\sin(angle(pq, xy)) \leq 4/s$$

The second construction

Proof.

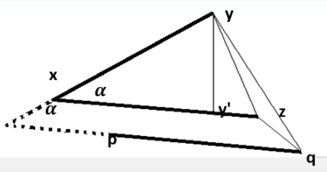
- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin\alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin\alpha \leq 4/s$



The second construction

Proof.

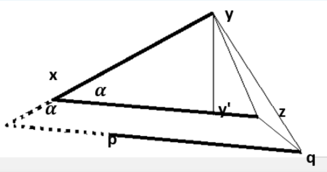
- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin\alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin\alpha \leq 4/s$



The second construction

Proof.

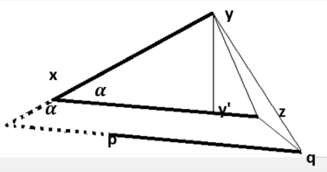
- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin \alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin \alpha \leq 4/s$



The second construction

Proof.

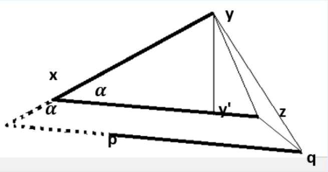
- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin\alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin\alpha \leq 4/s$



The second construction

Proof.

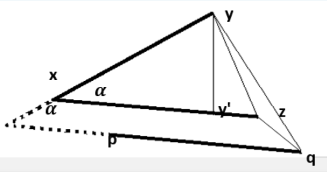
- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin\alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin\alpha \leq 4/s$



The second construction

Proof.

- If $s \leq 4$, then the claim clearly holds. assume that $s > 4$.
- Let $\alpha = angle(pq, xy)$. Let l be the ray emanating from x that is parallel to pq.
- $\sin\alpha = \frac{|yy'|}{|xy|} \leq \frac{|yz|}{|xy|} \leq \frac{|yq| + |qz|}{|xy|} = \frac{|yq| + |px|}{|xy|}$
- Since A and B are well-separated, Lemma 9.1.2:
- $|yq| \leq (2/s)|xy|$ and $|pq| \leq (2/s)|xy|$
- This implies that $\sin\alpha \leq 4/s$



□

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

□

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_ib_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_ib_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

The second construction

Lemma 10.1.7

The graph $G = (S, E)$ is a t-spanner for S.

Proof.

- We will show that the edges of E satisfy the condition of Lemma 10.1.4. First observe that, by Lemma 10.1.5:
- $0 < w < (\cos\theta - \sin\theta)/2$ and
- $t \geq 1/(\cos\theta - \sin\theta - 2w)$
- Let p and q be any two distinct points of S, and let i be the integer such that (i)$p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$.
- Assume (i) holds.
- Consider the representatives a_i and b_i of A_i and B_i ,respectively.
- Then $\sin(angle(pq, a_i b_i)) \leq 4/s \leq \sin\theta/2$ hence,
- $angle(pq, a_i b_i) \leq \theta/2$

□

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \le |a_i b_i|$
- $angle(pq, xy) \le angle(pq, a_i b_i) + angle(a_i b_i, xy) \le \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \le (1 + 4/s)|pq|$
- $|a_i b_i| \le |pq|/\cos\theta$
- in particular $|xy| \le |pq|/\cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq|/\cos\theta$
- in particular $|xy| \leq |pq|/\cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq|/\cos\theta$
- in particular $|xy| \leq |pq|/\cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \le |a_i b_i|$
- $angle(pq, xy) \le angle(pq, a_i b_i) + angle(a_i b_i, xy) \le \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \le (1 + 4/s)|pq|$
- $|a_i b_i| \le |pq|/\cos\theta$
- in particular $|xy| \le |pq|/\cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq| / \cos\theta$
- in particular $|xy| \leq |pq| / \cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq| / \cos\theta$
- in particular $|xy| \leq |pq| / \cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq| / \cos\theta$
- in particular $|xy| \leq |pq| / \cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq|/\cos\theta$
- in particular $|xy| \leq |pq|/\cos\theta$

The second construction

Proof.

- Let C be the cone of C_k such that $(a_i, b_i) \in E_0(C)$
- Recall that u_i is the node of the split tree T for which $s_u = A_i$. It is clear that u_i is on the path from the root to the leaf storing p.
- Therefore, when we considered the cone C and the point p, (a_i, b_i) was one of the edges that satisfied conditions 1. and 2. above.
- Let (x, y) be the edges of $E_0(C)$ that was marked when we considered C and p.
- Then (x, y) is an edges of E, and $|xy| \leq |a_i b_i|$
- $angle(pq, xy) \leq angle(pq, a_i b_i) + angle(a_i b_i, xy) \leq \theta/2 + \theta/2 = \theta$
- Since A_i and B_i are well-separated , $|a_i b_i| \leq (1 + 4/s)|pq|$
- $|a_i b_i| \leq |pq| / \cos\theta$
- in particular $|xy| \leq |pq| / \cos\theta$

□

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_i$ and $y \in B_i$,or (ii) $y \in A_i$ and $x \in B_i$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_i$.
- Since A_j and B_j are well-separated $|pq| \leq (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

□

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_i$ and $y \in B_i$,or (ii) $y \in A_i$ and $x \in B_i$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_i$.
- Since A_j and B_j are well-separated $|pq| \leq (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_j$ and $y \in B_j$,or (ii) $y \in A_j$ and $x \in B_j$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_j$.
- Since A_j and B_j are well-separated $|pq| \le (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_i$ and $y \in B_i$,or (ii) $y \in A_i$ and $x \in B_i$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_i$.
- Since A_j and B_j are well-separated $|pq| \leq (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_i$ and $y \in B_i$,or (ii) $y \in A_i$ and $x \in B_i$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_i$.
- Since A_j and B_j are well-separated $|pq| \leq (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

The second construction

Proof.

- Finally, let j be the integer such that (i) $x \in A_i$ and $y \in B_i$,or (ii) $y \in A_i$ and $x \in B_i$
- Assume (i) holds.
- When we considered the cone C and the point p, we marked (x, y)
- Therefore, u_j is on the path in T from the root to the leaf storing p and, hence, $p \in A_i$.
- Since A_j and B_j are well-separated $|pq| \leq (2/S)|xy| = w|xy|$
- Since all the premises of Lemma 10.1.4 are satisfied, we have proved that G is a t-spanner.

□

The second construction

Lemma 10.1.8

The out degree of each vertex of the graph $G = (S, E)$ is less than or equal to $2|C_k| = O(1/\theta^{d-1})$

Proof.

- Let x be a point of S, and let C be a cone of C_k . Clearly, it is sufficient to show that, in G, x is the source of at most two edges in cone C.
- Assume that the edge set E contains three pairwise distinct edges (x, y),(x, y') and $(x, y")$ that are all contained in $E_0(C)$ We may assume without loss of generality that $|xy| \leq |xy'| \leq |xy"|$
- Moreover, in case $|xy| = |xy'|$, we may assume without loss of generality that (x, y) has a lower index than (x, y'). similarly in case $|xy'| = |xy"|$ we may assume without loss of generality that (x, y') has a lower index than $(x, y")$

□

The second construction

Lemma 10.1.8

The out degree of each vertex of the graph $G = (S, E)$ is less than or equal to $2|C_k| = O(1/\theta^{d-1})$

Proof.

- Let x be a point of S, and let C be a cone of C_k . Clearly, it is sufficient to show that, in G, x is the source of at most two edges in cone C.
- Assume that the edge set E contains three pairwise distinct edges (x, y),(x, y') and $(x, y")$ that are all contained in $E_0(C)$ We may assume without loss of generality that $|xy| \leq |xy'| \leq |xy"|$
- Moreover, in case $|xy| = |xy'|$, we may assume without loss of generality that (x, y) has a lower index than (x, y'). similarly in case $|xy'| = |xy"|$ we may assume without loss of generality that (x, y') has a lower index than $(x, y")$

The second construction

Lemma 10.1.8

The out degree of each vertex of the graph $G = (S, E)$ is less than or equal to $2|C_k| = O(1/\theta^{d-1})$

Proof.

- Let x be a point of S, and let C be a cone of C_k . Clearly, it is sufficient to show that, in G, x is the source of at most two edges in cone C.
- Assume that the edge set E contains three pairwise distinct edges (x, y),(x, y') and $(x, y")$ that are all contained in $E_0(C)$ We may assume without loss of generality that $|xy| \le |xy'| \le |xy"|$
- Moreover, in case $|xy| = |xy'|$, we may assume without loss of generality that (x, y) has a lower index than (x, y'). similarly in case $|xy'| = |xy"|$ we may assume without loss of generality that (x, y') has a lower index than $(x, y")$

The second construction

Lemma 10.1.8

The out degree of each vertex of the graph $G = (S, E)$ is less than or equal to $2|C_k| = O(1/\theta^{d-1})$

Proof.

- Let x be a point of S, and let C be a cone of C_k . Clearly, it is sufficient to show that, in G, x is the source of at most two edges in cone C.
- Assume that the edge set E contains three pairwise distinct edges (x, y),(x, y') and $(x, y")$ that are all contained in $E_0(C)$ We may assume without loss of generality that $|xy| \leq |xy'| \leq |xy"|$
- Moreover, in case $|xy| = |xy'|$, we may assume without loss of generality that (x, y) has a lower index than (x, y'). similarly in case $|xy'| = |xy"|$ we may assume without loss of generality that (x, y') has a lower index than $(x, y")$

□

The second construction

Proof.

- Let p be $\in S$, such that marked edge (x, y) when considered the cone C and the point p. Consider nodes u and v of T such that $(x_u, y_v) = (x, y)$
- u is on the path from the root to the leaf storing p. Define the point p of S, and the nodes u and v of T similarly with respect to the edge (x,y).Assume that u $= u'$. Then u is on the path from the root to the leaf storing p . Since $|xy| \leq |xy|$, could not have marked edge (x, y) when considered C and p,which is a contradiction. Therefore, $u' \neq u$. By symmetric arguments, it follows that $u' \neq u$" and u" $\neq u$. Hence the set
- $\{u_i \in T : 1 \leq i \leq m, a_i = x\} \cup \{v_i \in T : 1 \leq i \leq m, b_i = x\}$
- contains at least three nodes. This contradicts Lemma 10.1.1

The second construction

Proof.

- Let p be $\in S$, such that marked edge (x, y) when considered the cone C and the point p. Consider nodes u and v of T such that $(x_u, y_v) = (x, y)$
- u is on the path from the root to the leaf storing p. Define the point p of S, and the nodes u and v of T similarly with respect to the edge (x,y).Assume that u $= u'$. Then u is on the path from the root to the leaf storing p . Since $|xy| \leq |xy|$, could not have marked edge (x, y) when considered C and p,which is a contradiction. Therefore, $u' \neq u$. By symmetric arguments, it follows that $u' \neq u"$ and $u" \neq u$. Hence the set
- $\{u_i \in T : 1 \leq i \leq m, a_i = x\} \cup \{v_i \in T : 1 \leq i \leq m, b_i = x\}$
- contains at least three nodes. This contradicts Lemma 10.1.1

The second construction

Proof.

- Let p be $\in S$, such that marked edge (x, y) when considered the cone C and the point p. Consider nodes u and v of T such that $(x_u, y_v) = (x, y)$
- u is on the path from the root to the leaf storing p. Define the point p of S, and the nodes u and v of T similarly with respect to the edge (x,y).Assume that u $= u'$. Then u is on the path from the root to the leaf storing p . Since $|xy| \leq |xy|$, could not have marked edge (x, y) when considered C and p,which is a contradiction. Therefore, $u' \neq u$. By symmetric arguments, it follows that $u' \neq u$" and $u" \neq u$. Hence the set
- $\{u_i \in T : 1 \leq i \leq m, a_i = x\} \cup \{v_i \in T : 1 \leq i \leq m, b_i = x\}$
- contains at least three nodes. This contradicts Lemma 10.1.1

The second construction

Proof.

- Let p be $\in S$, such that marked edge (x, y) when considered the cone C and the point p. Consider nodes u and v of T such that $(x_u, y_v) = (x, y)$
- u is on the path from the root to the leaf storing p. Define the point p of S, and the nodes u and v of T similarly with respect to the edge (x,y).Assume that u $= u'$. Then u is on the path from the root to the leaf storing p . Since $|xy| \leq |xy|$, could not have marked edge (x, y) when considered C and p,which is a contradiction. Therefore, $u' \neq u$. By symmetric arguments, it follows that $u' \neq u"$ and $u" \neq u$. Hence the set
- $\{u_i \in T : 1 \leq i \leq m, a_i = x\} \cup \{v_i \in T : 1 \leq i \leq m, b_i = x\}$
- contains at least three nodes. This contradicts Lemma 10.1.1

□

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in C$
- for each cone C of C separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in C$
- for each cone C of C separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in \mathcal{C}$
- for each cone C of $\mathcal{C}$ separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in C$
- for each cone C of C separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in C$
- for each cone C of C separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

- assume that each edge $(x, y) \in E_0(C)$ has a pointer to the node u_i (or v_i) of the split tree T such that x is the representative of A_i (or B_i)

computing the edges of E

- compute the subsets $E_0(C)$, $C \in C$
- for each cone C of C separately do:
- Following these pointers, and store with each node e_u of T , a list L_u containing all edges $(x, y) \in E_0(C)$ such that x is the representative of S_u, for some pair $\{S_u, S_v\}$ in our WSPD.
- By considering all nodes u of T , compute the shortest edge, which denote by e_u,in the list L_u
- compute for each node u the shortest edge among all edges e_v ,where v ranges over the ancestors of u. Denote this shortest edge by $e_u{}'$

The second construction

Running Time

1. $O(nlogn + m)$, where $m = O(s^d n)$. the time for computing the split tree and the WSPD
2. $O(n + m) = O(m)$. the time for computing the graph $G_0 = (S, E_0)$
3. $O(1/\theta^{d-1} + mlog(1/\theta))$. the time for computing the subsets $E_0(C), C \in C$.
4. $O(n/\theta^{d-1} + m)$. the time for computing the set E of marked edges

The second construction

Running Time

1. $O(nlogn + m)$, where $m = O(s^d n)$. the time for computing the split tree and the WSPD
2. $O(n + m) = O(m)$. the time for computing the graph $G_0 = (S, E_0)$
3. $O(1/\theta^{d-1} + mlog(1/\theta))$. the time for computing the subsets $E_0(C), C \in C$.
4. $O(n/\theta^{d-1} + m)$. the time for computing the set E of marked edges

The second construction

Running Time

1. $O(nlogn + m)$, where $m = O(s^d n)$. the time for computing the split tree and the WSPD
2. $O(n + m) = O(m)$. the time for computing the graph $G_0 = (S, E_0)$
3. $O(1/\theta^{d-1} + mlog(1/\theta))$. the time for computing the subsets $E_0(C), C \in C$.
4. $O(n/\theta^{d-1} + m)$. the time for computing the set E of marked edges

The second construction

Running Time

1. $O(nlogn + m)$, where $m = O(s^d n)$. the time for computing the split tree and the WSPD
2. $O(n + m) = O(m)$. the time for computing the graph $G_0 = (S, E_0)$
3. $O(1/\theta^{d-1} + mlog(1/\theta))$. the time for computing the subsets $E_0(C), C \in C$.
4. $O(n/\theta^{d-1} + m)$. the time for computing the set E of marked edges

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \le i \le m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \leq i \leq m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \le i \le m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \le i \le m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \leq i \leq m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \le i \le m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \le i \le m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \leq i \leq m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let S be a set of n points in R^d
- Compute a split tree T for S
- Construct a WSPD with $s = 4(t+1)/(t-1)$
- $G = (S, E)$ where $E = \{\{a_i, b_i\}\} : 1 \leq i \leq m\}$, is a t-spanner for S (section 9.2)
- We will choose a_i , b_i from A_i, B_i wisely, using the split tree T.
- Let u be any internal node of T
- Let u_l and u_r be u left and right sons
- Let e_l and e_r be the edges that connect u to u_l and u_r respectively

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let n_l and n_r be number of leaves in the sub-trees rooted by n_l and n_r respectively.
- If $n_l \geq n_r$, we label e_l as heavy, and e_r as light.
- If $n_l < n_r$, we label e_r as heavy, and e_l as light.
- Each internal node is connected by exactly one heavy edge to one of its children.
- Each internal node has a unique chain of heavy edges down to the bottom of T.
- Let $l(u)$ be the leaf whose chain contains u.
- Let $r(u)$ be the point of S stored in $l(u)$.

Construction of Spanner with logarithmic diameter

- Let i be $1 \leq i \leq m$, the pair A_i , B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
- A_i,B_i are the set of points stored in the sub-trees of u_i and v_i
- We take $r(u_i) \in A_i$ to be its representative. We do the same for $r(v_i) \in B_i$.
- Now $G = (S, E)$ is a t-spanner. We need to prove its diameter is logarithmic.

Lemma 10.2.1

For each $i, 1 \leq i \leq m$, and for each point $p \in A_i$ there is a t-spanner path in G between p and the representative a_i of A_i, that contains at most $log|Ai|$ edges.

Construction of Spanner with logarithmic diameter

- Let i be $1 \leq i \leq m$, the pair A_i , B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
- A_i,B_i are the set of points stored in the sub-trees of u_i and v_i
- We take $r(u_i) \in A_i$ to be its representative. We do the same for $r(v_i) \in B_i$.
- Now $G = (S, E)$ is a t-spanner. We need to prove its diameter is logarithmic.

Lemma 10.2.1

For each $i, 1 \leq i \leq m$, and for each point $p \in A_i$ there is a t-spanner path in G between p and the representative a_i of A_i, that contains at most $log|Ai|$ edges.

Construction of Spanner with logarithmic diameter

- Let i be $1 \leq i \leq m$, the pair A_i , B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
- A_i,B_i are the set of points stored in the sub-trees of u_i and v_i
- We take $r(u_i) \in A_i$ to be its representative. We do the same for $r(v_i) \in B_i$.
- Now $G = (S, E)$ is a t-spanner. We need to prove its diameter is logarithmic.

Lemma 10.2.1

For each $i, 1 \leq i \leq m$, and for each point $p \in A_i$ there is a t-spanner path in G between p and the representative a_i of A_i, that contains at most $log|Ai|$ edges.

Construction of Spanner with logarithmic diameter

- Let i be $1 \leq i \leq m$, the pair A_i , B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
- A_i,B_i are the set of points stored in the sub-trees of u_i and v_i
- We take $r(u_i) \in A_i$ to be its representative. We do the same for $r(v_i) \in B_i$.
- Now $G = (S, E)$ is a t-spanner. We need to prove its diameter is logarithmic.

Lemma 10.2.1

For each $i, 1 \leq i \leq m$, and for each point $p \in A_i$ there is a t-spanner path in G between p and the representative a_i of A_i, that contains at most $log|Ai|$ edges.

Construction of Spanner with logarithmic diameter

- Let i be $1 \leq i \leq m$, the pair A_i , B_i is represented implicitly by 2 nodes of T, u_i and v_i, respectively.
- A_i,B_i are the set of points stored in the sub-trees of u_i and v_i
- We take $r(u_i) \in A_i$ to be its representative. We do the same for $r(v_i) \in B_i$.
- Now $G = (S, E)$ is a t-spanner. We need to prove its diameter is logarithmic.

Lemma 10.2.1

For each $i, 1 \leq i \leq m$, and for each point $p \in A_i$ there is a t-spanner path in G between p and the representative a_i of A_i, that contains at most $log|Ai|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

Construction of Spanner with logarithmic diameter

Proof.

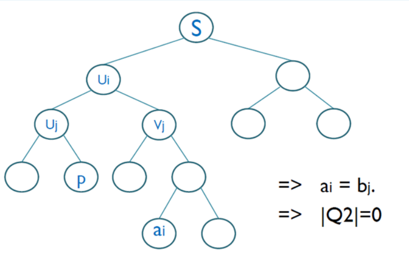
- From construction, there is a t-spanner path between p and a_i(Section 9.2).
- If $p = a_i$, we are done. Assume otherwise.Let j be the index such that
- (i) $p \in A_j$ and $a_i \in B_j$ Or (ii) $a_i \in A_j$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j
- Recursively construct a path Q_2 between a_i and its representative b_j
- Return $Q = Q_1, \{a_j, b_j\}, Q_2$.
- We will show that $|Q| \leq log|Ai|$, $(p \in A_i)$. Induction on the size of set A_i.
- Base: $A_i = \{a_i\} \longrightarrow p = a_i \longrightarrow$ we are done.
- Let i be an index such that $|A_i| > 1, p \neq a_i$
  Assume, for all k such that $|A_k| < |A_i|$, and for all $x \in A_k$, the algorithm constructs a t-spanner path between x and a_k with at most $log|Ak|$ edges.

□

Construction of Spanner with logarithmic diameter

Proof.

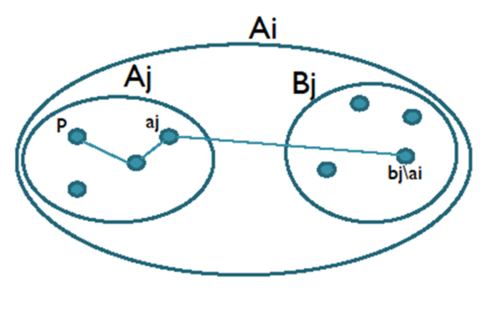
- Let u_i , u_j , v_j be the nodes of T whose sub-trees store the sets A_i, A_j, B_j respectively.
- u_i is a common ancestor of the leaves storing p and a_i.
- u_j lies on the path from root to the leaf of p.
- V_j lies on the path from root to the leaf of a_i
- A_j, B_j are disjoint $\longrightarrow$ u_j, v_j are both in the sub-tree of u_i, and neither is an ancestor of the other



Construction of Spanner with logarithmic diameter

Proof.

- $|Aj| \leq |Ai|/2$. Otherwise all the edges between u_i and u_j would be heavy.
- But then, the representative of A_i would be an element of A_j, but its not ($a_i \in B_j$ represents A_i).
- By induction hypothesis, the path Q_1 between p and a_j contains at most $log|A_j|$ edge, which is
- $\leq log|A_i| - 1$ We add $\{a_j, b_j\}$ and get that $|Q| \leq log|A_i|$



Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1$, $\{a_j, b_j\}$, Q_2.
- We proved that $|Q_1| \leq log|A_j|$.

□

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1$, $\{a_j, b_j\}$, Q_2.
- We proved that $|Q_1| \le log|A_j|$.

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1 \ , \ \{a_j, b_j\} \ , \ Q_2$.
- We proved that $|Q_1| \leq log|A_j|$.

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1$, $\{a_j, b_j\}$, Q_2.
- We proved that $|Q_1| \leq log|A_j|$.

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1$, $\{a_j, b_j\}$, Q_2.
- We proved that $|Q_1| \leq log|A_j|$.

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1 \ , \ \{a_j, b_j\} \ , \ Q_2$.
- We proved that $|Q_1| \leq log|A_j|$.

Construction of Spanner with logarithmic diameter

Lemma 10.2.2

G is a t-spanner for S, whose diameter is less than equal to $2logn - 1$

Proof.

- Let p and q be 2 distinct points in G. We will show that there is a t-spanner path between them with at most $2logn1$ edges
- Let j be the index such that:
  (i) $p \in A_j$ and $q \in B_j$. Or (ii) $q \in Aj$ and $p \in B_j$.
- Recursively construct a path Q_1 between p and its representative a_j.
- Recursively construct a path Q_2 between a_i and its representative b_j.
- Return $Q = Q_1$, $\{a_j, b_j\}$, Q_2.
- We proved that $|Q_1| \leq log|A_j|$.

□

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

Construction of Spanner with logarithmic diameter

Proof.

- Similarly, $|Q2| \leq log|B_j|$
- Hence, $|Q| \leq log|A_j| + log|B_j| + 1$.
- A_j, B_j are disjoint $\longrightarrow |A_j| + |B_j| \leq n$
- $log|A_j| + log|B_j| \leq log|A_j| + log|n - A_j| \leq 2log(n/2) = 2logn - 2$
- Hence, Q contains at most $2logn - 1$ edges

□

Time complexity

1. WSPD construction: $O(nlogn + s^d n)$ (Theorem 9.4.6).
2. Representative selection: $O(m)$.
3. All in all : $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

The Closest Pair Problem

- its possible to compute a WSPD t-spanner of $O(n)$ edges in $O(nLogn)$ time.(corollary 9.4.7)
- In a t-spanner where $1 \leq t \leq 2$ the smallest edge in the graph is always present in the t-spanner.
- By creating a t-spanner of $t = 2$, we know one of its edges is the smallest in the graph, and we can scan all its edges in $O(n)$
- find the closest pair in $O(nLogn) + O(n) = O(nLogn)$

Theorem 10.3.1

Given a set S of n points in R^d ,algorithm $ClosestPair(S)$ computes a closest pair in S in $O(nlogn)$ time.

Theorem 10.3.2

Any algebraic computation-tree algorithm that, when given a set S of n points in R^d and a real number $s > 2$, computes a WSPD for S with separation ratio s, has a worst-case running time of $O(nlogn)$.

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and $q_i \imath S$ is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq \ldots \leq |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and $q_i \iota S$ is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\}, ..., \{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq ... \leq |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and $q_i \imath S$ is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\}, ..., \{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq ... \leq |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and q_i ıS is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\},...,\{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq \ldots \leq |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \le k \le$
- A sequence of $\{p_i, q_i\}, 1 \le i \le k$, where $p_i \neq q_i$ and p_i and $q_i \imath S$ is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \le i \le k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \le |R(A_2)R(B_2)| \le \ldots \le |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and q_iıS is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\},...,\{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq ... \leq |R(A_m)R(B_m)|$

Computing K Closest Pair Problem

- Let S be a set of n points in R^d
- Let k be an integer such that $1 \leq k \leq$
- A sequence of $\{p_i, q_i\}, 1 \leq i \leq k$, where $p_i \neq q_i$ and p_i and $q_i \imath S$ is called a sequence of k closest pair if the distance of $|p_i q_i|$, $1 \leq i \leq k$ are the smallest k edges in S.
- Reminder: in WSPD where $X \in R^d$, $R(X)$ is the smallest binding box of all the points in X.
- Let $s > 0$ be a real number, consider the split tree T, and the corresponding WSPD:$\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$
- Where $m = O(s^d n)$
- We shall mark the minimum distance between $R(A_i)$ and $R(B_i)$ as $|R(A_i)R(B_i)|$, and lets assume with out the loss of generality that:$|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq \ldots \leq |R(A_m)R(B_m)|$

Algorithm

Algorithm KClosestPairs(S, k)

Comment: This algorithm takes as input a set S of n points in $\mathbb{R}^d$ and an integer k such that $1 \le k \le \binom{n}{2}$. It assumes that a WSPD for S has been computed already. The algorithm returns a sequence of k closest pairs in S.

Step 1: Compute the smallest integer $\ell \ge 1$, such that

$$\sum_{i=1}^{\ell} |A_i| \cdot |B_i| \ge k.$$

Step 2: Let $r := |R(A_\ell)R(B_\ell)|$. Compute the integer ℓ', which is defined as the number of indices i with $1 \le i \le m$, such that

$$|R(A_i)R(B_i)| \le (1 + 4/s)r.$$

Compute the set L consisting of all pairs $\{p, q\}$ for which there is an index i with $1 \le i \le \ell'$, such that $p \in A_i$ and $q \in B_i$, or $q \in A_i$ and $p \in B_i$.

Step 3: Compute and return the k smallest distances determined by the pairs in the set L.

K Closest Pairs Correctness

Lemma 10.3.3

Let p, q be two distinct points in S, and let j be the index such that $p \in A_j$ and $q \in B_j$ or $q \in A_j$ and $p \in B_j$ if $j > l'$ the $\{p, q\}$ is not one of the k closest pairs of the set S.

Proof.

- For any index $i, 1 \leq i \leq m$ Let $x_i \in R(A_i)$ and $y_i \in R(B_i)$ such that $|x_i y_i| = |R(A_i)R(B_i)|$ (in general x_i,y_i not points of s)
- Let l be any index such that $1 \leq i \leq l$, let a be any point of Ai, and let b any point of B_i.

K Closest Pairs Correctness

Lemma 10.3.3

Let p, q be two distinct points in S, and let j be the index such that $p \in A_j$ and $q \in B_j$ or $q \in A_j$ and $p \in B_j$ if $j > l'$ the $\{p, q\}$ is not one of the k closest pairs of the set S.

Proof.

- For any index $i, 1 \leq i \leq m$ Let $x_i \in R(A_i)$ and $y_i \in R(B_i)$ such that $|x_i y_i| = |R(A_i)R(B_i)|$ (in general x_i,y_i not points of s)
- Let l be any index such that $1 \leq i \leq l$, let a be any point of Ai, and let b any point of B_i.

□

Proof.

- $|ab| \leq (1 + 4/s)|R(A_i)R(B_i)| \leq (1 + 4/s)|R(A_l)R(B_l)| = (1 + 4/s)r$
- By step 1 of the algo.The pairs $\{A_i, B_i\}$ $1 \leq i \leq l$ satisfy: $(\sum |A_i||B_i|) \geq k$
- And thus determine at least k distances, all which are less than or equal to $(1 + 4/s)r$
- On the other hand, since, we have:$j > l'$

$$|pq| \geq |R(A_j)R(B_j)| \geq (1 + 4/s)r$$

□

Proof.

- $|ab| \leq (1 + 4/s)|R(A_i)R(B_i)| \leq (1 + 4/s)|R(A_l)R(B_l)| = (1 + 4/s)r$
- By step 1 of the algo.The pairs $\{A_i, B_i\}$ $1 \leq i \leq l$ satisfy: $(\sum |A_i||B_i|) \geq k$
- And thus determine at least k distances, all which are less than or equal to $(1 + 4/s)r$
- On the other hand, since, we have:$j > l'$

$$|pq| \geq |R(A_j)R(B_j)| \geq (1 + 4/s)r$$

□

Proof.

- $|ab| \leq (1 + 4/s)|R(A_i)R(B_i)| \leq (1 + 4/s)|R(A_l)R(B_l)| = (1 + 4/s)r$
- By step 1 of the algo.The pairs $\{A_i, B_i\}$ $1 \leq i \leq l$ satisfy: $(\sum |A_i||B_i|) \geq k$
- And thus determine at least k distances, all which are less than or equal to $(1 + 4/s)r$
- On the other hand, since, we have:$j > l'$

$$|pq| \geq |R(A_j)R(B_j)| \geq (1 + 4/s)r$$

Proof.

- $|ab| \le (1 + 4/s)|R(A_i)R(B_i)| \le (1 + 4/s)|R(A_l)R(B_l)| = (1 + 4/s)r$
- By step 1 of the algo.The pairs $\{A_i, B_i\}$ $1 \le i \le l$ satisfy: $(\sum |A_i||B_i|) \ge k$
- And thus determine at least k distances, all which are less than or equal to $(1 + 4/s)r$
- On the other hand, since, we have:$j > l'$

$$|pq| \ge |R(A_j)R(B_j)| \ge (1 + 4/s)r$$

$\square$

K Closest Pairs - runtime

- Step 1: because of the usage of the split tree, by traversing it in post order, we can compute the number of leaves for each node and can compare for each $1 \le i \le m$ the value of $|A_i||B_i|$, in $O(m)$, hence Step 1 takes $O(m)$
- Step 2: each node of the split tree stores the bounding box of the points in the subtree, there for l' can be computed in $O(m)$. Given l', L can be computed in:$O(\sum |A_i||B_i|)$
- Step 3: because k is a constant, we can find the $k`th$ smallest distance in linear time, than we can select all the pairs, which distance is smaller then or equal to the kth
- Conclusion: $O(m) + O(\sum |A_i||B_i|) = O(m + \sum |A_i||B_i|)$

K Closest Pairs - runtime

- Step 1: because of the usage of the split tree, by traversing it in post order, we can compute the number of leaves for each node and can compare for each $1 \leq i \leq m$ the value of $|A_i||B_i|$, in $O(m)$, hence Step 1 takes $O(m)$
- Step 2: each node of the split tree stores the bounding box of the points in the subtree, there for l' can be computed in $O(m)$. Given l', L can be computed in:$O(\sum |A_i||B_i|)$
- Step 3: because k is a constant, we can find the $k`th$ smallest distance in linear time, than we can select all the pairs, which distance is smaller then or equal to the kth
- Conclusion: $O(m) + O(\sum |A_i||B_i|) = O(m + \sum |A_i||B_i|)$

K Closest Pairs - runtime

- Step 1: because of the usage of the split tree, by traversing it in post order, we can compute the number of leaves for each node and can compare for each $1 \leq i \leq m$ the value of $|A_i||B_i|$, in $O(m)$, hence Step 1 takes $O(m)$
- Step 2: each node of the split tree stores the bounding box of the points in the subtree, there for l' can be computed in $O(m)$. Given l', L can be computed in: $O(\sum |A_i||B_i|)$
- Step 3: because k is a constant, we can find the $k`th$ smallest distance in linear time, than we can select all the pairs, which distance is smaller then or equal to the kth
- Conclusion: $O(m) + O(\sum |A_i||B_i|) = O(m + \sum |A_i||B_i|)$

K Closest Pairs - runtime

- Step 1: because of the usage of the split tree, by traversing it in post order, we can compute the number of leaves for each node and can compare for each $1 \leq i \leq m$ the value of $|A_i||B_i|$, in $O(m)$, hence Step 1 takes $O(m)$
- Step 2: each node of the split tree stores the bounding box of the points in the subtree, there for l' can be computed in $O(m)$. Given l', L can be computed in:$O(\sum |A_i||B_i|)$
- Step 3: because k is a constant, we can find the k'th smallest distance in linear time, than we can select all the pairs, which distance is smaller then or equal to the kth
- Conclusion:
  $O(m) + O(\sum |A_i||B_i|) = O(m + \sum |A_i||B_i|)$

K Closest Pairs - runtime

Lemma 10.3.4

Let δ be the k-th smallest distance in the set S and let $r = |A_l||B_l|$ be the value computed in Step 2, then $r \leq \delta$

Proof.

- Lets assume $r > \delta$, so for any $i \geq l$

$$|pq| \geq |R(A_i)R(B_i)| \geq |R(A_l)R(B_l)| = r > \delta$$

- Contradicts the build of Step 1 (to choose the min l value).

K Closest Pairs - runtime

Lemma 10.3.4

Let δ be the k-th smallest distance in the set S and let $r = |A_l||B_l|$ be the value computed in Step 2, then $r \leq \delta$

Proof.

- Lets assume $r > \delta$, so for any $i \geq l$

$$|pq| \geq |R(A_i)R(B_i)| \geq |R(A_l)R(B_l)| = r > \delta$$

- Contradicts the build of Step 1 (to choose the min l value).

$\square$

K Closest Pairs - runtime

Lemma 10.3.5

Let v be the kth smallest distance in S, and let $\{p, q\}$ be any pair of points that is contained in L, then:$|pq| \leq (1 + 4/s)^2 v$

Proof.

- $|pq| \leq (1 + 4/s)|x_i y_i|$
- $|R(A_i)R(B_i)| \leq (1 + 4/s)r$
- $|pq| \leq (1 + 4/s)|x_i y_i| = (1 + 4/s)|R(A_i)R(B_i)| \leq (1 + 4/s)^2 r \leq (1 + 4/s)^2 v$

K Closest Pairs - runtime

Lemma 10.3.5

Let v be the kth smallest distance in S, and let $\{p, q\}$ be any pair of points that is contained in L, then:$|pq| \le (1 + 4/s)^2 v$

Proof.

- $|pq| \le (1 + 4/s)|x_i y_i|$
- $|R(A_i)R(B_i)| \le (1 + 4/s)r$
- $|pq| \le (1 + 4/s)|x_i y_i| = (1 + 4/s)|R(A_i)R(B_i)| \le (1 + 4/s)^2 r \le (1 + 4/s)^2 v$

K Closest Pairs - runtime

Lemma 10.3.5

Let v be the kth smallest distance in S, and let $\{p, q\}$ be any pair of points that is contained in L, then:$|pq| \leq (1 + 4/s)^2 v$

Proof.

- $|pq| \leq (1 + 4/s)|x_i y_i|$
- $|R(A_i)R(B_i)| \leq (1 + 4/s)r$
- $|pq| \leq (1 + 4/s)|x_i y_i| = (1 + 4/s)|R(A_i)R(B_i)| \leq (1 + 4/s)^2 r \leq (1 + 4/s)^2 v$

$\square$

K Closest Pairs - runtime

- Let M denote the number of distances that are less then or equal to $(1+4/s)^2 v$. Then the total running time of the algorithm is $O(n+M)$.
- Full running time:
- WSPD for S of size $m = O(n)$ can be calculated in $O(nLogn)$
- The algo. Runs at $O(m+M)$
- $O(nLogn) + O(n+n+2k) = O(nLogn+k)$

K Closest Pairs - runtime

- Let M denote the number of distances that are less then or equal to $(1+4/s)^2 v$. Then the total running time of the algorithm is $O(n+M)$.
- Full running time:
- WSPD for S of size $m = O(n)$ can be calculated in $O(nLogn)$
- The algo. Runs at $O(m+M)$
- $O(nLogn) + O(n+n+2k) = O(nLogn+k)$

K Closest Pairs - runtime

- Let M denote the number of distances that are less then or equal to $(1+4/s)^2 v$. Then the total running time of the algorithm is $O(n+M)$.
- Full running time:
- WSPD for S of size $m = O(n)$ can be calculated in $O(nLogn)$
- The algo. Runs at $O(m+M)$
- $O(nLogn) + O(n+n+2k) = O(nLogn+k)$

K Closest Pairs - runtime

- Let M denote the number of distances that are less then or equal to $(1+4/s)^2 v$. Then the total running time of the algorithm is $O(n+M)$.
- Full running time:
- WSPD for S of size $m = O(n)$ can be calculated in $O(nLogn)$
- The algo. Runs at $O(m+M)$
- $O(nLogn) + O(n+n+2k) = O(nLogn+k)$

K Closest Pairs - runtime

- Let M denote the number of distances that are less then or equal to $(1+4/s)^2 v$. Then the total running time of the algorithm is $O(n+M)$.
- Full running time:
- WSPD for S of size $m = O(n)$ can be calculated in $O(nLogn)$
- The algo. Runs at $O(m+M)$
- $O(nLogn) + O(n+n+2k) = O(nLogn+k)$

All-nearest neighbors

- In this problem with are given the set S, and want to compute the nearest neighbor for each point in S.
- In this analysis we will use a generalization of the fact that any point can be the nearest neighbor of at most a constant number of other points.

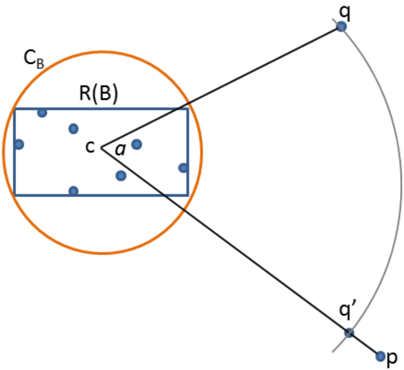
All-nearest neighbors

- In this problem with are given the set S, and want to compute the nearest neighbor for each point in S.
- In this analysis we will use a generalization of the fact that any point can be the nearest neighbor of at most a constant number of other points.

All-nearest neighbors

Lemma 10.3.7

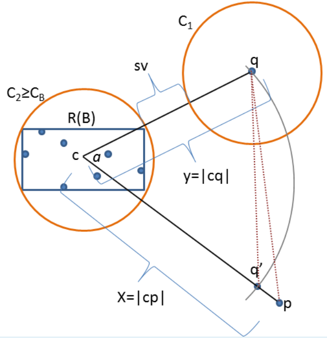
Let B be a finite set of points.Let C_B be the smallest ball that contains the bounding box $R(B)$ of B. Let c be the center of C_B .Let $s > 1$ be a real number. Let p,q be 2 distinct points in R^d such that both $\{p\}$ and B, and $\{q\}$ and B are well-seperated with respect to s. Assume that $|pC_B| \leq |pq|$ and $|qC_B| \leq |pq|$. Let $\alpha = angle(cp, cq)$. Then we have $\alpha \geq (s-1)/s$



All-nearest neighbors

Proof.

- If $\alpha \geq 1$ then obviously $(s-1)/s \leq 1$
- If $\alpha < 1$, Let r be the radius of C_B
- Let $x = |pc|$ and $y = |qc|$, lets assumed without loss of generality that $x \geq y$.
- Let q' be point on the line $|cp|$, such that $|q'c| = |qc|$

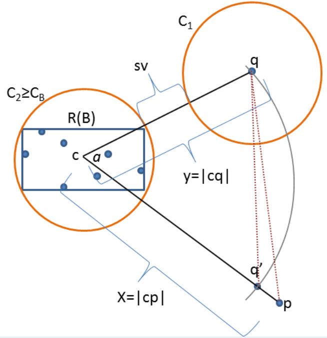


All-nearest neighbors

Proof.

- By using Triangle inequality we recieve:$|pq| \leq |pq'| + |q'q| \leq (x - y) + \alpha y = x - (1 - \alpha)y \ *$
- Since $\{q\}$ and B are well separated, lets draw 2 balls, C_1(around Q), and C_2 (around $R(B)$) with radius v, there for $|C_1 C_2| \geq sv$. Since C_B is the smallest ball of B, $v \geq r$, then we can conclude:

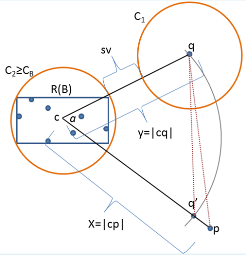
$$y = |qc| \geq |C_1 C_2| \geq sv \geq sr \ **$$



All-nearest neighbors

Proof.

- By using $*$, $**$ and the fact that $\alpha < 1$ we get: $|pq| \leq x - (1-\alpha)y \leq x - (1-\alpha)sr(***)$
- Using the same method as $*$ we claim: $x = |pc| \geq sr$
- Since $s > 1$, then p is out of the ball CB , $x = |pc| = |pC_B| + r \leq |pq| + r$
- Combined with $***$ we receive: $|pq| \leq (|pq| + r) - (1-\alpha)sr$
- Which is in fact: $\alpha \geq (s-1)/s$



All-nearest neighbors

Lemma 10.3.8

Let A and B be to finite sets of points, let C_B be the smallest ball to contain $R(B)$, let $s > 1$ be a real number. $\forall p \in A$ the sets $\{p\}$ and B are well seperated with respect to s.Assume $|pC_B| \leq |pq|$ for any p,q that belong to A.Then the set A contains $O((s/(s-1))d)$ elements.

Proof.

- Using Lemma 10.3.7, we know that: $angle(cp, cq) \geq (s-1)/s$
- Using this equation and a theorem which bounds the size of any set of points for which the minimum angle is at least some given real number we prove that A contains: $O((s/(s-1))^d)$ elements

□

All-nearest neighbors

- For any node u in the split tree T, S_u denotes the set of all points of S that are stored in the subtree of u
- We define $F(u)$ to be the set of all points $p \in S$ such that the pair $\{\{p\}, S_u\}$ is contained in some ancestor of u.
- We define $N(u)$ to be the set of all points $p \in F(u)$ such that, the distance from p to the smallest ball containing $R(S_u)$ is less than or equal to the smallest distance between p and any other point of $F(u)$
- Obeserve: $N(u) \subseteq F(u)$

All-nearest neighbors

- For any node u in the split tree T, S_u denotes the set of all points of S that are stored in the subtree of u
- We define $F(u)$ to be the set of all points $p \in S$ such that the pair $\{\{p\}, S_u\}$ is contained in some ancestor of u.
- We define $N(u)$ to be the set of all points $p \in F(u)$ such that, the distance from p to the smallest ball containing $R(S_u)$ is less than or equal to the smallest distance between p and any other point of $F(u)$
- Obeserve:$N(u) \subseteq F(u)$

All-nearest neighbors

- For any node u in the split tree T, S_u denotes the set of all points of S that are stored in the subtree of u
- We define $F(u)$ to be the set of all points $p \in S$ such that the pair $\{\{p\}, S_u\}$ is contained in some ancesstor of u.
- We define $N(u)$ to be the set of all points $p \in F(u)$ such that, the distance from p to the smallest ball containing $R(S_u)$ is less than or equal to the smallest distance between p and any other point of $F(u)$
- Obeserve: $N(u) \subseteq F(u)$

All-nearest neighbors

- For any node u in the split tree T, S_u denotes the set of all points of S that are stored in the subtree of u
- We define $F(u)$ to be the set of all points $p \in S$ such that the pair $\{\{p\}, S_u\}$ is contained in some ancestor of u.
- We define $N(u)$ to be the set of all points $p \in F(u)$ such that, the distance from p to the smallest ball containing $R(S_u)$ is less than or equal to the smallest distance between p and any other point of $F(u)$
- Obeserve: $N(u) \subseteq F(u)$

All-nearest neighbors

Lemma 10.3.9

For any node u of T, the size of $N(u)$ is $O((s/(s-1))d)$

Proof.

- Let $A = N(u)$ and $B = S_u$
- We will show that those two sets satisfy the conditions of lemma 10.3.8.
- Let $p \in A$, then $p \in F(u)$ hence, there is an ancestor v of u, such that $\{p\}$ and S_v are well separated. Since S_u is a subset of S_v the sets $\{p\}$ and $S_u = B$ are well separated as well.
- Let p and q be two distinct points of A. Then by the definition of $N(u)$, the distance between p and C_B is less then or equal to the smallest distance between p and any other point of $F(u)$. In particular since $q \in F(u)$, we have $|pC_B| \le |pq|$.

□

All-nearest neighbors

Lemma 10.3.9

For any node u of T, the size of $N(u)$ is $O((s/(s-1))d)$

Proof.

- Let $A = N(u)$ and $B = S_u$
- We will show that those two sets satisfy the conditions of lemma 10.3.8.
- Let $p \in A$, then $p \in F(u)$ hence, there is an ancestor v of u, such that $\{p\}$ and S_v are well separated. Since S_u is a subset of S_v the sets $\{p\}$ and $S_u = B$ are well separated as well.
- Let p and q be two distinct points of A. Then by the definition of $N(u)$, the distance between p and C_B is less then or equal to the smallest distance between p and any other point of $F(u)$. In particular since $q \in F(u)$, we have $|pC_B| \le |pq|$.

□

All-nearest neighbors

Lemma 10.3.9

For any node u of T, the size of $N(u)$ is $O((s/(s-1))d)$

Proof.

- Let $A = N(u)$ and $B = S_u$
- We will show that those two sets satisfy the conditions of lemma 10.3.8.
- Let $p \in A$, then $p \in F(u)$ hence, there is an ancestor v of u, such that $\{p\}$ and S_v are well separated. Since S_u is a subset of S_v the sets $\{p\}$ and $S_u = B$ are well separated as well.
- Let p and q be two distinct points of A. Then by the definition of $N(u)$, the distance between p and C_B is less then or equal to the smallest distance between p and any other point of $F(u)$. In particular since $q \in F(u)$, we have $|pC_B| \leq |pq|$.

All-nearest neighbors

Lemma 10.3.9

For any node u of T, the size of $N(u)$ is $O((s/(s-1))d)$

Proof.

- Let $A = N(u)$ and $B = S_u$
- We will show that those two sets satisfy the conditions of lemma 10.3.8.
- Let $p \in A$, then $p \in F(u)$ hence, there is an ancestor v of u, such that $\{p\}$ and S_v are well separated. Since S_u is a subset of S_v the sets $\{p\}$ and $S_u = B$ are well separated as well.
- Let p and q be two distinct points of A. Then by the definition of $N(u)$, the distance between p and C_B is less then or equal to the smallest distance between p and any other point of $F(u)$. In particular since $q \in F(u)$, we have $|pC_B| \leq |pq|$.

□

All-nearest neighbors

- Let p be the point of S, let q be the nearest neighbor of p, and let u be the leaf of T that contains q. We know that there is an index i, such that $A_i = \{p\}$ and $q \in B_i$ (or vise versa).
- Hence, there is an ancestor v or u, such that $B_i = S_v$
- Therefore,$p \in F(u)$, also since $S_u = q$, the distance between p and the smallest ball containing $R(S_u)$ is just $|pq|$, which is clearly less then or equal to the distance between p and any other point in $F(u)$, this proves:
- $p \in N(u)$

All-nearest neighbors

- Let p be the point of S, let q be the nearest neighbor of p, and let u be the leaf of T that contains q. We know that there is an index i, such that $A_i = \{p\}$ and $q \in B_i$ (or vise versa).
- Hence, there is an ancestor v or u, such that $B_i = S_v$
- Therefore,$p \in F(u)$, also since $S_u = q$, the distance between p and the smallest ball containing $R(S_u)$ is just $|pq|$, which is clearly less then or equal to the distance between p and any other point in $F(u)$, this proves:
- $p \in N(u)$

All-nearest neighbors

- Let p be the point of S, let q be the nearest neighbor of p, and let u be the leaf of T that contains q. We know that there is an index i, such that $A_i = \{p\}$ and $q \in B_i$ (or vise versa).
- Hence, there is an ancestor v or u, such that $B_i = S_v$
- Therefore,$p \in F(u)$, also since $S_u = q$, the distance between p and the smallest ball containing $R(S_u)$ is just $|pq|$, which is clearly less then or equal to the distance between p and any other point in $F(u)$, this proves:
- $p \in N(u)$

All-nearest neighbors

- Let p be the point of S, let q be the nearest neighbor of p, and let u be the leaf of T that contains q. We know that there is an index i, such that $A_i = \{p\}$ and $q \in B_i$ (or vise versa).
- Hence, there is an ancestor v or u, such that $B_i = S_v$
- Therefore,$p \in F(u)$, also since $S_u = q$, the distance between p and the smallest ball containing $R(S_u)$ is just $|pq|$, which is clearly less then or equal to the distance between p and any other point in $F(u)$, this proves:
- $p \in N(u)$

All-nearest neighbors

- Let p be the point of S, let q be the nearest neighbor of p, and let u be the leaf of T that contains q. We know that there is an index i, such that $A_i = \{p\}$ and $q \in B_i$ (or vise versa).
- Hence, there is an ancestor v or u, such that $B_i = S_v$
- Therefore,$p \in F(u)$, also since $S_u = q$, the distance between p and the smallest ball containing $R(S_u)$ is just $|pq|$, which is clearly less then or equal to the distance between p and any other point in $F(u)$, this proves:
- $p \in N(u)$

Algorithm Runtime:O(nlogn)

Algorithm ALLNEARESTNEIGHBORS(S)

Comment: This algorithm takes as input a set S of n points in $\mathbb{R}^d$. It returns for each point in S its nearest neighbor.

Step 1: Choose a constant $s > 2$, and compute the split tree T and the corresponding WSPD for S, with respect to s, having size $O(n)$. O(nlogn)

Step 2: Compute the sets $N(u)$, for all nodes u of T. O(n)

Step 3: Compute a list L consisting of all pairs $\{p, q_u\}$ of points, where u ranges over all leaves of the split tree, q_u is the point of S stored at u, and p ranges over all points of $N(u)$. O(n)

Step 4: For each $p \in S$, compute a point q_p such that $\{p, q_p\} \in L$ and $|pq_p|$ is minimum. O(n)

Step 5: Return the pairs $\{p, q_p\}$, where p ranges over all points of S.

Computing an approximate minimum spanning tree

Algorithm ApproxMST(S, t)

Comment: This algorithm takes as input a set S of n points in $\mathbb{R}^d$, and a real number $t > 1$. It returns a t-approximate minimum spanning tree of S.

Step 1: Compute the WSPD t-spanner G of Corollary 9.4.7.

Step 2: Using algorithm Prim(G), see Section 2.6.2, compute a minimum spanning tree T of G.

Step 3: Return the tree T.