Chapter 20

Well-Separated Pair Decomposition

By Sariel Har-Peled, March 23, 2023[1]

> The fact remains that getting people right is not what living is all about anyway. It's getting them wrong that is living, getting them wrong and wrong and wrong and then, on careful reconsideration, getting them wrong again. That's how we known we're alive: we're wrong. Maybe the best thing would be to forget being right or wrong about people and just go along for the ride. But if you can do that - well, lucky you.
>
> American Pastoral, Philip Roth

In this chapter, we will investigate how to represent distances between points efficiently. Naturally, an explicit description of the distances between n points requires listing all the $\binom{n}{2}$ distances. Here we will show that there is a considerably more compact representation which is sufficient if all we care about are approximate distances. This representation would have many nice applications.

20.1. Well-separated pair decomposition (WSPD)

Let P be a set of n points in $\mathbb{R}^{\mathsf{d}}$, and let $1/4 > \varepsilon > 0$ be a parameter. One can represent all distances between points of P by explicitly listing the $\binom{n}{2}$ pairwise distances. Of course, the listing of the coordinates of each point gives us an alternative, more compact representation (of size $\mathsf{d}n$), but its not a very informative representation. We are interested in a representation that will capture the structure of the distances between the points.

As a concrete example, consider the three points on the right. We would like to have a representation that captures the fact that p has similar distance to q and s, and furthermore, that q and s are close together as far as p is concerned. As such, if we are interested in the closest pair among the three points, we will only check the distance between q and s, since they are the only pair (among the three) that might realize the closest pair.

Figure 20.1.1

Denote by $A \otimes B = \left\{ \{x, y\} \,\middle|\, x \in A,\, y \in B, x \neq y \right\}$ the set of all the (unordered) pairs of points formed by the sets A and B. We will be informal and refer to $A \otimes B$ as a ***pair*** of the sets A and B. Here, we are interested in schemes that cover all possible pairs of points of P by a small collection of such pairs.

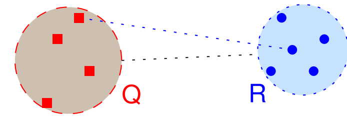
Definition 20.1.1 (Pair decomposition). For a point set P, a ***pair decomposition*** of P is a set of pairs

$$\mathcal{W} = \left\{ \{A_1, B_1\}, \ldots, \{A_s, B_s\} \right\},$$

such that (I) $A_i, B_i \subset \mathsf{P}$ for every i, (II) $A_i \cap B_i = \emptyset$ for every i, and (III) $\bigcup_{i=1}^{s} A_i \otimes B_i = \mathsf{P} \otimes \mathsf{P}$.

Translation: For any pair of distinct points $p, q \in \mathsf{P}$, there is at least one (and usually exactly one) pair $\{A_i, B_i\} \in \mathcal{W}$ such that $p \in A_i$ and $q \in B_i$.

Definition 20.1.2. The pair Q and R is ***$(1/\varepsilon)$-separated*** if



$$\max(\operatorname{diam}(\mathsf{Q}), \operatorname{diam}(\mathsf{R})) \le \varepsilon \cdot \mathsf{d}(\mathsf{Q}, \mathsf{R}),$$

where $\mathsf{d}(\mathsf{Q}, \mathsf{R}) = \min_{q \in \mathsf{Q}, \mathsf{s} \in \mathsf{R}} \|q\mathsf{s}\|$.

Intuitively, the pair $\mathsf{Q} \otimes \mathsf{R}$ is ***$(1/\varepsilon)$-separated*** if all the points of Q have roughly the same distance to the points of R. Alternatively, imagine covering the two point sets with two balls of minimum size, and require that the distance between the two balls is at least $2/\varepsilon$ times the radius of the larger of the two.

Thus, for the three points of Figure 4.1.1, the pairs $\{p\} \otimes \{q, \mathsf{s}\}$ and $\{q\} \otimes \{\mathsf{s}\}$ are (say) 2-separated and describe all the distances among these three points. (The gain here is quite marginal, as we replaced the distance description, made out of three pairs of points, by the distance between two pairs of sets. But stay tuned – exciting things are about to unfold.)

Motivated by the above example, a well-separated pair decomposition is a way to describe a metric by such "well-separated" pairs of sets.

Definition 20.1.3 (WSPD). For a point set P, a ***well-separated pair decomposition*** (***WSPD***) of P with parameter $1/\varepsilon$ is a pair decomposition of P with a set of pairs

$$\mathcal{W} = \Big\{ \{A_1, B_1\}, \ldots, \{A_s, B_s\} \Big\},$$

such that, for any i, the sets A_i and B_i are ε^{-1}-separated.

For a concrete example of a WSPD, see Figure 4.1.2.

Instead of maintaining such a decomposition explicitly, it is convenient to construct a tree $\mathcal{T}$ having the points of P as leaves. Now every pair (A_i, B_i) is just a pair of nodes (v_i, u_i) of $\mathcal{T}$, such that $A_i = \mathsf{P}_{v_i}$ and $B_i = \mathsf{P}_{u_i}$, where P_v denotes the points of P stored in the subtree of v (here v is a node of $\mathcal{T}$). Naturally, in our case, the tree we would use is a compressed quadtree of P, but any tree that decomposes the points such that the diameter of a point set stored in a node drops quickly as we go down the tree might work. Naturally, even when the underlying tree is specified, there are many possible WSPDs that can be represented using this tree. Naturally, we will try to find a WSPD that is "minimal".

This WSPD representation using a tree gives us a compact representation of the distances of the point set.

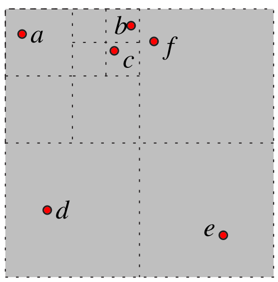
Corollary 20.1.4. *For an ε^{-1}-WSPD $\mathcal{W}$, it holds, for any pair $\{u, v\} \in \mathcal{W}$, that*

$$\forall q \in \mathsf{P}_u, \mathsf{s} \in \mathsf{P}_v \quad \max\Big(\operatorname{diam}(\mathsf{P}_u), \operatorname{diam}(\mathsf{P}_v)\Big) \le \varepsilon \, \|q\mathsf{s}\| \,.$$

It would usually be convenient to associate with each set P_u in the WSPD an arbitrary representative point $\mathrm{rep}_u \in \mathsf{P}$. Selecting and assigning these representative points can always be done by a simple **DFS** traversal of the tree used to represent the WSPD.

20.1.1. The construction algorithm

Given the point set P in $\mathbb{R}^{\mathsf{d}}$, the algorithm first computes the compressed quadtree $\mathcal{T}$ of P. Next, the algorithm works by being greedy. It tries to put into the WSPD pairs of nodes in the tree that are as high as possible. In particular, if a pair $\{u, v\}$ would be generated, then the pair formed by the parents



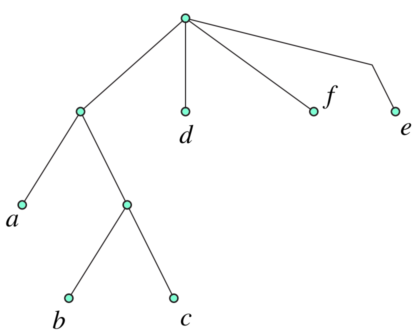
(i)

$$
\begin{gathered}
A_1 = \{d\},\ B_1 = \{e\}\\
A_2 = \{a, b, c\},\ B_2 = \{e\}\\
A_3 = \{a, b, c\},\ B_3 = \{d\}\\
A_4 = \{a\},\ B_4 = \{b, c\}\\
A_5 = \{b\},\ B_5 = \{c\}\\
A_6 = \{a\},\ B_6 = \{f\}\\
A_7 = \{b\},\ B_7 = \{f\}\\
A_8 = \{c\},\ B_8 = \{f\}\\
A_9 = \{d\},\ B_9 = \{f\}\\
A_{10} = \{e\},\ B_{10} = \{f\}
\end{gathered}
$$

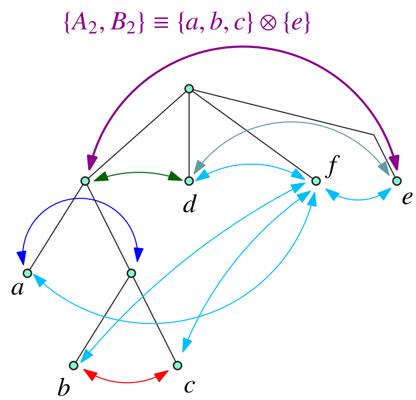
(ii)

$$
\mathcal{W} = \left\{
\begin{array}{c}
\{A_1, B_1\},\\
\{A_2, B_2\},\\
\{A_3, B_3\},\\
\{A_4, B_4\},\\
\{A_5, B_5\},\\
\{A_6, B_6\},\\
\{A_7, B_7\},\\
\{A_8, B_8\},\\
\{A_9, B_9\},\\
\{A_{10}, B_{10}\}
\end{array}
\right\}
$$

(iii)



(iv)



(v)

Figure 20.1.2: (i) A point set $\mathsf{P} = \{a, b, c, d, e, f\}$. (ii) The decomposition into pairs. (iii) The respective $(1/2)$-WSPD. For example, the pair of points b and e (and their distance) is represented by $\{A_2, B_2\}$ as $b \in A_2$ and $e \in B_2$. (iv) The quadtree T representing the point set P. (v) The WSPD as defined by pairs of vertices of T.

of this pair of nodes will not be well separated. As such, the algorithm starts from the root and tries to separate it from itself. If the current pair is not well separated, then we replace the bigger node of the pair by its children (i.e., thus replacing a single pair by several pairs). Clearly, sooner or later this refinement process would reach well-separated pairs, which it would output. Since it considers all possible distances up front (i.e., trying to separate the root from itself), it would generate a WSPD covering all pairs of points.

Definition 20.1.5. Let $\Delta(v)$ denote the ***diameter*** of the cell associated with a node v of the quadtree $\mathcal{T}$. We tweak this definition a bit so that if a node contains a single point or is empty, then it is zero. This would make our algorithm easier to describe.

Formally, $\Delta(v) = 0$ if P_v is either empty or a single point. Otherwise, it is the diameter of the region associated with v; that is, $\Delta(v) = \operatorname{diam}(\Box_v)$, where $\Box_v$ (we remind the reader) is the quadtree cell associated with the node v. Note that since $\mathcal{T}$ is a compressed quadtree, we can always decide if $|\mathsf{P}_v| > 1$ by just checking if the subtree rooted at v has more than one node (since then this subtree must store more than one point).

We define the ***geometric distance*** between two nodes u and v of $\mathcal{T}$ to be

$$\mathbf{d}(u,v) = \mathbf{d}(\Box_u, \Box_v) = \min_{p \in \Box_u, q \in \Box_v} \|pq\| .$$

We compute the compressed quadtree $\mathcal{T}$ of P in $O(n \log n)$ time. Next, we compute the WSPD by calling **algWSPD**$(u_0, u_0, \mathcal{T})$, where u_0 is the root of $\mathcal{T}$ and **algWSPD** is depicted in Figure 4.1.3.

```
algWSPD(u, v)
   if u = v and Δ(u) = 0 then
        return          // Do not pair a leaf with itself
   if Δ(u) < Δ(v) then
        Exchange u and v
   If Δ(u) ≤ ε · d(u, v) then
        return   {{u, v}}

   // u_1, ..., u_r – the children of u
   return ⋃_{i=1}^{r} algWSPD(u_i, v).
```

Figure 20.1.3: The algorithm **algWSPD** for computing well-separated pair decomposition. The nodes u and v belong to a compressed quadtree $\mathcal{T}$ of P.

20.1.1.1. Analysis

The following lemma is implied by an easy packing argument.

Lemma 20.1.6. *Let $\Box$ be a cell of a grid G of $\mathbb{R}^{\mathsf{d}}$ with cell diameter x. For $y \geq x$, the number of cells in G at distance at most y from $\Box$ is $O\Big((y/x)^{\mathsf{d}}\Big)$. (The $O(\cdot)$ notation here, and in the rest of the chapter, hides a constant that depends exponentially on d.)*

Lemma 20.1.7. **algWSPD** *terminates and computes a valid pair decomposition.*

Proof: By induction, it follows that every pair of points of P is covered by a pair of subsets $\{\mathsf{P}_u, \mathsf{P}_v\}$ output by the **algWSPD** algorithm. Note that **algWSPD** always stops if both u and v are leafs, which implies that **algWSPD** always terminates.

Now, observe that if $\{u, v\}$ is in the output pair and either P_u or P_v is not a single point, then $\alpha = \max(\mathrm{diam}(\mathsf{P}_u), \mathrm{diam}(\mathsf{P}_v)) > 0$. This implies that $\mathbf{d}(\mathsf{P}_u, \mathsf{P}_v) \geq \mathbf{d}(u, v) \geq \Delta(u)/\varepsilon \geq \alpha/\varepsilon > 0$. Namely, $\mathsf{P}_u \cap \mathsf{P}_v = \emptyset$. ∎

Lemma 20.1.8. *For the WSPD generated by* **algWSPD**, *we have that for any pair $\{u, v\}$ in the WSPD,*

$$\max\Big(\mathrm{diam}(\mathsf{P}_u), \mathrm{diam}(\mathsf{P}_v)\Big) \leq \varepsilon \cdot \mathbf{d}(u, v) \quad \textit{and} \quad \mathbf{d}(u, v) \leq \|q\mathsf{s}\|$$

hold for any $q \in \mathsf{P}_u$ and $\mathsf{s} \in \mathsf{P}_v$.

Proof: For every output pair $\{u, v\}$, we have by the design of the algorithm that

$$\max\Big(\mathrm{diam}(\mathsf{P}_u), \mathrm{diam}(\mathsf{P}_v)\Big) \leq \max\Big(\Delta(u), \Delta(v)\Big) \leq \varepsilon \cdot \mathbf{d}(u, v).$$

Also, for any $q \in \mathsf{P}_u$ and $\mathsf{s} \in \mathsf{P}_v$, we have $\mathbf{d}(u, v) = \mathbf{d}(\square_u, \square_v) \leq \mathbf{d}(\mathsf{P}_u, \mathsf{P}_v) \leq \mathsf{d}(q, \mathsf{s})$, since $\mathsf{P}_u \subseteq \square_u$ and $\mathsf{P}_v \subseteq \square_v$. ∎

Lemma 20.1.9. *For a pair $\{u, v\} \in \mathcal{W}$, computed by* **algWSPD**, *we have that*

$$\max\Big(\Delta(u),\ \Delta(v)\Big) \leq \min\Big(\Delta(\overline{\mathrm{p}}(u)),\ \Delta(\overline{\mathrm{p}}(v))\Big),$$

where $\overline{\mathrm{p}}(x)$ denotes the parent node of the node x in the tree $\mathcal{T}$.

Proof: We trivially have that $\Delta(u) < \Delta(\overline{\mathrm{p}}(u))$ and $\Delta(v) < \Delta(\overline{\mathrm{p}}(v))$. The idea is to track the pairs used in the recursive calls to generate the pair $\{u, v\}$, see Figure 4.1.4.

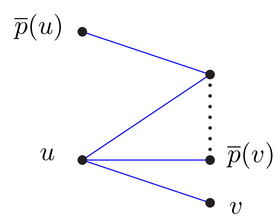


Figure 20.1.4: The y-order encodes the sidelengths of the cells, higher up means bigger cells.

A pair $\{u, v\}$ is generated because of a sequence of recursive calls **algWSPD**(u_0, u_0), **algWSPD**(u_1, v_1), ..., **algWSPD**(u_s, v_s), where $u_s = u$, $v_s = v$, and u_0 is the root of $\mathcal{T}$. Assume that $u_{s-1} = u$ and $v_{s-1} = \overline{\mathrm{p}}(v)$. Then $\Delta(u) \leq \Delta(\overline{\mathrm{p}}(v))$, since the algorithm always refines the larger cell.

Similarly, let t be the last index such that $u_{t-1} = \overline{\mathrm{p}}(u)$ (namely, $u_{t-1} \neq u_t = u$ and $v_{t-1} = v_t$). Then, since v is a descendant of v_{t-1}, it holds that

$$\Delta(v) \leq \Delta(v_t) = \Delta(v_{t-1}) \leq \Delta(u_{t-1}) = \Delta(\overline{\mathrm{p}}(u)),$$

since (again) the algorithm always refines the larger cell in the pair $\{u_{t-1}, v_{t-1}\}$. ∎

Lemma 20.1.10. *The number of pairs in the computed WSPD is $O(n/\varepsilon^{\mathsf{d}})$.*

Proof: Let $\{u, v\}$ be a pair appearing in the output. Consider the sequence (i.e., stack) of recursive calls that led to this output. In particular, assume that the last recursive call to **algWSPD**(u, v) was issued by **algWSPD**(u, v'), where $v' = \overline{\mathrm{p}}(v)$ is the parent of v in $\mathcal{T}$. Then

$$\Delta(\overline{\mathrm{p}}(u)) \geq \Delta(v') \geq \Delta(u),$$

by Lemma 4.1.9.

We charge the pair $\{u, v\}$ to the node v' and claim that each node of $\mathcal{T}$ is charged at most $O\Big(\varepsilon^{-\mathsf{d}}\Big)$ times. To this end, fix a node $v' \in \mathsf{V}(\mathcal{T})$, where $\mathsf{V}(\mathcal{T})$ is the set of vertices of $\mathcal{T}$. Since the pair $\{u, v'\}$ was not output by **algWSPD** (despite being considered), we conclude that $\Delta(v') > \varepsilon \cdot \mathbf{d}(u, v')$ and as such $\mathbf{d}(u, v') < r = \Delta(v')/\varepsilon$. Now, there are several possibilities:

(i) $\Delta(v') = \Delta(u)$. But there are at most $O\Big((r/\Delta(v'))^{\mathsf{d}}\Big) = O(1/\varepsilon^{\mathsf{d}})$ nodes that have the same level (i.e., diameter) as v' such that their cells are within a distance at most r from it, by Lemma 4.1.6. Thus, this type of charge can happen at most $O(2^{\mathsf{d}} \cdot (1/\varepsilon^{\mathsf{d}}))$ times, since v' has at most 2^{d} children.

(ii) $\Delta(\overline{\mathrm{p}}(u)) = \Delta(v')$. By the same argumentation as above $\mathbf{d}(\overline{\mathrm{p}}(u), v') \leq \mathbf{d}(u, v') < r$. There are at most $O(1/\varepsilon^{\mathsf{d}})$ such nodes $\overline{\mathrm{p}}(u)$. Since the node $\overline{\mathrm{p}}(u)$ has at most 2^{d} children, it follows that the number of such charges is at most $O\Big(2^{\mathsf{d}} \cdot 2^{\mathsf{d}} \cdot (1/\varepsilon^{\mathsf{d}})\Big)$.

(iii) $\Delta(\overline{\mathrm{p}}(u)) > \Delta(v') > \Delta(u)$. Consider the canonical grid G having $\square_{v'}$ as one of its cells (see Definition 4.6.2). Let $\widehat{\square}$ be the cell in G containing $\square_u$. Observe that $\square_u \subsetneq \widehat{\square} \subsetneq \square_{\overline{\mathrm{p}}(u)}$. In addition, $\mathbf{d}\Big(\widehat{\square}, \square_{v'}\Big) \leq \mathbf{d}(\square_u, \square_{v'}) = \mathbf{d}(u, v') < r$. It follows that there are at most $O(1/\varepsilon^{\mathsf{d}})$ cells like $\widehat{\square}$ that might participate in charging v', and as such, the total number of charges is $O(2^{\mathsf{d}}/\varepsilon^{\mathsf{d}})$, as claimed.

As such, v' can be charged at most $O\Big(2^{2d}/\varepsilon^{\mathsf{d}}\Big) = O\Big(1/\varepsilon^{\mathsf{d}}\Big)$ times[②]. This implies that the total number of pairs generated by the algorithm is $O\Big(n\varepsilon^{-\mathsf{d}}\Big)$, since the number of nodes in $\mathcal{T}$ is $O(n)$. ∎

Since the running time of **algWSPD** is clearly linear in the output size, we have the following result.

Theorem 20.1.11. *For $1 \geq \varepsilon > 0$, and a set P of n points in $\mathbb{R}^d$, one can construct, in $O\Big(n \log n + n\varepsilon^{-\mathsf{d}}\Big)$. time, an ε^{-1}-WSPD of P of size $n\varepsilon^{-\mathsf{d}}$.*

20.2. Applications of WSPD

20.2.1. Spanners

It is sometime beneficial to describe distances between n points by using a sparse graph to encode the distances.

Definition 20.2.1. For a weighted graph G and any two vertices p and q of G, we will denote by $\mathsf{d}_{\mathsf{G}}(q, \mathsf{s})$ the ***graph distance*** between q and s. Formally, $\mathsf{d}_{\mathsf{G}}(q, \mathsf{s})$ is the length of the shortest path in G between q and s. It is easy to verify that d_{G} is a metric; that is, it complies with the triangle inequality. Naturally, if q and s belongs to two different connected components of G, then $\mathsf{d}_{\mathsf{G}}(q, \mathsf{s}) = \infty$.

[②]We remind the reader that we will usually consider the dimension d to be a constant, and the O notation would happily consume any constants that depend only on d. Conceptually, you can think about the O as being a black hole for such constants, as its gravitational force tears such constants away. The shrieks of horror of these constants as they are being swallowed alive by the black hOle can be heard every time you look at the O.

Such a graph might be useful algorithmically if it captures the distances we are interested in while being sparse (i.e., having few edges). In particular, if such a graph G over n vertices has only $O(n)$ edges, then it can be manipulated efficiently, and it is a compact implicit representation of the $\binom{n}{2}$ distances between all the pairs of vertices of G.

A ***t-spanner*** of a set of points $\mathsf{P} \subset \mathbb{R}^{\mathsf{d}}$ is a weighted graph G whose vertices are the points of P, and for any $q, \mathsf{s} \in \mathsf{P}$, we have

$$\|q\mathsf{s}\| \leq \mathsf{d}_{\mathsf{G}}(q, \mathsf{s}) \leq t\, \|q\mathsf{s}\|.$$

The ratio $\mathsf{d}_{\mathsf{G}}(q, \mathsf{s}) / \|q\mathsf{s}\|$ is the ***stretch*** of q and s in G. The ***stretch*** of G is the maximum stretch of any pair of points of P.

20.2.1.1. Construction

We are given a set of n points in $\mathbb{R}^{\mathsf{d}}$ and a parameter $1 \geq \varepsilon > 0$. We will construct a spanner as follows.

Let $c \geq 16$ be an arbitrary constant, and set $\delta = \varepsilon / c$. Compute a δ^{-1}-WSPD decomposition of P using the algorithm of Theorem 4.1.11. For any vertex u in the quadtree $\mathcal{T}$ (used in computing the WSPD), let rep_u be an arbitrary point of P_u. For every pair $\{u, v\} \in \mathcal{W}$, add an edge between $\{\mathrm{rep}_u, \mathrm{rep}_v\}$ with weight $\|\mathrm{rep}_u \mathrm{rep}_v\|$, and let G be the resulting graph.

One can prove that the resulting graph is connected and that it contains all the points of P. We will not prove this explicitly as this is implied by the analysis below.

20.2.1.2. Analysis

Observe that by the triangle inequality, we have that $\mathsf{d}_{\mathsf{G}}(q, \mathsf{s}) \geq \|q\mathsf{s}\|$, for any $q, \mathsf{s} \in \mathsf{P}$.

Theorem 20.2.2. *Given a set* P *of* n *points in* $\mathbb{R}^{\mathsf{d}}$ *and a parameter* $1 \geq \varepsilon > 0$*, one can compute a* $(1+\varepsilon)$*-spanner of* P *with* $O\!\left(n\varepsilon^{-\mathsf{d}}\right)$ *edges, in* $O\!\left(n \log n + n\varepsilon^{-\mathsf{d}}\right)$ *time.*

Proof: The construction is described above. The upper bound on the stretch is proved by induction on the length of pairs in the WSPD. So, fix a pair $x, y \in \mathsf{P}$, and assume that by the induction hypothesis, for any pair $z, w \in \mathsf{P}$ such that $\|zw\| < \|xy\|$, it follows that $\mathbf{d}_{\mathsf{G}}(z, w) \leq (1+\varepsilon)\, \|zw\|$.

The pair x, y must appear in some pair $\{u, v\} \in \mathcal{W}$, where $x \in \mathsf{P}_u$ and $y \in \mathsf{P}_v$. Thus, by construction

$$\|\mathrm{rep}_u \mathrm{rep}_v\| \leq \mathbf{d}(u, v) + \Delta(u) + \Delta(v) \leq (1+2\delta)\mathbf{d}(u, v) \leq (1+2\delta)\, \|xy\| \tag{20.2.1}$$

and this implies

$$\begin{aligned} \max(\|\mathrm{rep}_u x\|, \|\mathrm{rep}_v - y\|) &\leq \max(\Delta(u), \Delta(v)) \leq \delta \cdot \mathbf{d}(u, v) \leq \delta\, \|\mathrm{rep}_u - \mathrm{rep}_v\| \\ &\leq \delta(1+2\delta)\, \|x - y\| < \frac{1}{4}\, \|x - y\|, \end{aligned}$$

by Theorem 4.1.11 and since $\delta \leq 1/16$. As such, we can apply the induction hypothesis (twice) to the pairs of points rep_u, x and rep_v, y, implying that

$$\mathsf{d}_{\mathsf{G}}(x, \mathrm{rep}_u) \leq (1+\varepsilon)\, \|\mathrm{rep}_u - x\| \qquad \text{and} \qquad \mathsf{d}_{\mathsf{G}}(\mathrm{rep}_v, y) \leq (1+\varepsilon)\, \|y - \mathrm{rep}_v\|.$$

Now, since $\mathrm{rep}_u \mathrm{rep}_v$ is an edge of G, $\mathsf{d}_{\mathsf{G}}(\mathrm{rep}_u, \mathrm{rep}_v) \leq \|\mathrm{rep}_u - \mathrm{rep}_v\|$. Thus, by the inductive hypothesis, the triangle inequality and Eq. (4.2.1), we have that

$$\|x - y\| \leq \mathsf{d}_{\mathsf{G}}(x, y) \leq \mathsf{d}_{\mathsf{G}}(x, \mathrm{rep}_u) + \mathsf{d}_{\mathsf{G}}(\mathrm{rep}_u, \mathrm{rep}_v) + \mathsf{d}_{\mathsf{G}}(\mathrm{rep}_v, y)$$

$$
\begin{aligned}
&\leq (1+\varepsilon)\left\|\mathrm{rep}_u - x\right\| + \left\|\mathrm{rep}_u - \mathrm{rep}_v\right\| + (1+\varepsilon)\left\|\mathrm{rep}_v - y\right\| \\
&\leq 2(1+\varepsilon)\cdot\delta\cdot\left\|\mathrm{rep}_u - \mathrm{rep}_v\right\| + \left\|\mathrm{rep}_u - \mathrm{rep}_v\right\| \\
&\leq (1+2\delta+2\varepsilon\delta)\left\|\mathrm{rep}_u - \mathrm{rep}_v\right\| \leq (1+2\delta+2\varepsilon\delta)\,(1+2\delta)\left\|x-y\right\| \\
&\leq (1+\varepsilon)\left\|x-y\right\|.
\end{aligned}
$$

The last step follows by an easy calculation. Indeed, since $c \geq 16$ and $c\delta = \varepsilon \leq 1$, we have that

$$
(1+2\delta+2\varepsilon\delta)(1+2\delta) \leq (1+4\delta)(1+2\delta) = 1+6\delta+8\delta^2 \leq 1+14\delta \leq 1+\varepsilon,
$$

as required. ∎

20.2.2. Approximating the minimum spanning tree

For a graph G, let $\mathsf{G}_{\leq r}$ denote the subgraph of G resulting from removing all the edges of weight (strictly) larger than r from G.

Lemma 20.2.3. *Given a set P of n points in $\mathbb{R}^{\mathsf{d}}$, one can compute a spanning tree T of P, such that $\omega(\mathsf{T}) \leq (1+\varepsilon)\omega(\mathit{MST})$, where MST is a minimum spanning tree of P and $\omega(\mathsf{T})$ is the total weight of the edges of T. This takes $O\left(n\log n + n\varepsilon^{-\mathsf{d}}\right)$ time. Furthermore, for any $r \geq 0$ and a connected component C of $\mathit{MST}_{\leq r}$, the set C is contained in a connected component of $\mathsf{T}_{\leq(1+\varepsilon)r}$.*

Proof: Compute a $(1+\varepsilon)$-spanner G of P and let T be a minimum spanning tree of G. We output the tree T as the approximate minimum spanning tree. Clearly the time to compute T is $O\left(n\log n + n\varepsilon^{-\mathsf{d}}\right)$, since the MST of a graph with n vertices and m edges can be computed in $O(n\log n + m)$ time.

There remains the task of proving that T is the required approximation. For any $q, \mathsf{s} \in \mathsf{P}$, let $\pi_{q\mathsf{s}}$ denote the shortest path between q and s in G. Since G is a $(1+\varepsilon)$-spanner, we have that $w(\pi_{q\mathsf{s}}) \leq (1+\varepsilon)\left\|q-\mathsf{s}\right\|$, where $w(\pi_{q\mathsf{s}})$ denotes the weight of $\pi_{q\mathsf{s}}$ in G. We have that $\mathsf{G}' = (\mathsf{P}, E)$ is a connected subgraph of G, where

$$
E = \bigcup_{q\mathsf{s}\in E(\mathsf{MST})} \pi_{uv},
$$

where $E(\mathsf{MST})$ denotes the set of edges of the graph MST. Furthermore,

$$
\omega(\mathsf{G}') \leq \sum_{q\mathsf{s}\in\mathsf{MST}} w(\pi_{q\mathsf{s}}) \leq \sum_{q\mathsf{s}\in\mathsf{MST}} (1+\varepsilon)\left\|q-\mathsf{s}\right\| = (1+\varepsilon)\omega(\mathsf{MST}),
$$

since G is a $(1+\varepsilon)$-spanner. Since G' is a connected spanning subgraph of G, it follows that $\omega(\mathsf{T}) \leq \omega(\mathsf{G}') \leq (1+\varepsilon)w(\mathsf{MST})$.

The second claim follows by similar argumentation. ∎

20.2.3. Approximating the diameter

Lemma 20.2.4. *Given a set P of n points in $\mathbb{R}^{\mathsf{d}}$, one can compute, in $O\left(n\log n + n\varepsilon^{-\mathsf{d}}\right)$ time, a pair $p, q \in \mathsf{P}$, such that $\left\|pq\right\| \geq (1-\varepsilon)\mathrm{diam}(\mathsf{P})$, where $\mathrm{diam}(\mathsf{P})$ is the diameter of P.*

Proof: Compute a $(4/\varepsilon)$-WSPD $\mathcal{W}$ of P. As before, we assign for each node u of $\mathcal{T}$ an arbitrary representative point that belongs to P_u. This can be done in linear time. Next, for each pair of $\mathcal{W}$, compute the distance of its representative points (for each pair, this takes constant time). Let $\{x, y\}$ be the pair in $\mathcal{W}$ such that the distance between its two representative points is maximal, and return rep_x and

rep_y as the two points realizing the approximation. Overall, this takes $O\Big(n \log n + n\varepsilon^{-\mathsf{d}}\Big)$ time, since $|\mathcal{W}| = O(n/\varepsilon^{\mathsf{d}})$.

To see why it works, consider the pair $q, \mathsf{s} \in \mathsf{P}$ realizing the diameter of P, and let $\{u, v\} \in \mathcal{W}$ be the pair in the WSPD that contain the two points, respectively (i.e., $q \in \mathsf{P}_u$ and $\mathsf{s} \in \mathsf{P}_v$). We have that

$$\begin{aligned}\|\mathrm{rep}_u - \mathrm{rep}_v\| &\geq \mathbf{d}(u, v) \geq \|q - \mathsf{s}\| - \mathrm{diam}(\mathsf{P}_u) - \mathrm{diam}(\mathsf{P}_v)\\ &\geq (1 - 2(\varepsilon/2))\,\|q - \mathsf{s}\| = (1 - \varepsilon)\mathrm{diam}(\mathsf{P}),\end{aligned}$$

since, by Corollary 4.1.4, $\max(\mathrm{diam}(\mathsf{P}_u), \mathrm{diam}(\mathsf{P}_v)) \leq 2(\varepsilon/4)\,\|q - \mathsf{s}\|$. Namely, the distance of the two points output by the algorithm is at least $(1 - \varepsilon)\mathrm{diam}(\mathsf{P})$. ∎

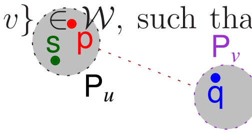
20.2.4. Closest pair

Let P be a set of points in $\mathbb{R}^{\mathsf{d}}$. We would like to compute the ***closest pair***, namely, the two points closest to each other in P.

We need the following observation.

Lemma 20.2.5. *Let $\mathcal{W}$ be an ε^{-1}-WSPD of P, for $\varepsilon \leq 1/2$. There exists a pair $\{u, v\} \in \mathcal{W}$, such that*
(i) $|\mathsf{P}_u| = |\mathsf{P}_v| = 1$ and
(ii) $\|\mathrm{rep}_u \mathrm{rep}_v\|$ is the length of the closest pair, where $\mathsf{P}_u = \{\mathrm{rep}_u\}$ and $\mathsf{P}_v = \{\mathrm{rep}_v\}$.

Proof: Consider the pair of closest points p and q in P, and consider the pair $\{u, v\} \in \mathcal{W}$, such that $p \in \mathsf{P}_u$ and $q \in \mathsf{P}_v$. If P_u contains an additional point $\mathsf{s} \in \mathsf{P}_u$, then we have that



$$\|p\mathsf{s}\| \leq \mathrm{diam}(\mathsf{P}_u) \leq \varepsilon \cdot \mathbf{d}(u, v) \leq \varepsilon\,\|p - q\| < \|p - q\|\,,$$

by Theorem 4.1.11 and since $\varepsilon = 1/2$. Thus, $\|p\mathsf{s}\| < \|pq\|$, a contradiction to the choice of p and q as the closest pair. Thus, $|\mathsf{P}_u| = |\mathsf{P}_v| = 1$ and $\mathrm{rep}_u = q$ and $\mathrm{rep}_v = \mathsf{s}$. ∎

Algorithm. Compute an ε^{-1}-WSPD $\mathcal{W}$ of P, for $\varepsilon = 1/2$. Next, scan all the pairs of $\mathcal{W}$, and compute for all the pairs $\{u, v\}$ which connect singletons (i.e., $|\mathsf{P}_u| = |\mathsf{P}_v| = 1$) the distance between their representatives rep_u and rep_v. The algorithm returns the closest pair of points encountered.

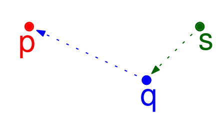
Theorem 20.2.6. *Given a set P of n points in $\mathbb{R}^{\mathsf{d}}$, one can compute the closest pair of points of P in $O(n \log n)$ time.*

We remind the reader that we already saw a linear (expected) time algorithm for this problem. However, this is a deterministic algorithm, and it can be applied in more abstract settings where a small WSPD still exists, while the previous algorithm would not work.

20.2.5. All nearest neighbors

Given a set P of n points in $\mathbb{R}^{\mathsf{d}}$, we would like to compute for each point $q \in \mathsf{P}$ its ***nearest neighbor*** in P (formally, this is the closest point in $\mathsf{P} \setminus \{q\}$ to q).

This is harder than it might seem at first, since this is *not* a symmetrical relationship. Indeed, in the figure on the right, q is the nearest neighbor to p, but s is the nearest neighbor to q.



20.2.5.1. The bounded spread case

Algorithm. Assume P is contained in the unit square and $\operatorname{diam}(\mathsf{P}) \geq 1/4$. Furthermore, let $\Phi = \Phi(\mathsf{P})$ denote the spread of P. Compute an ε^{-1}-WSPD $\mathcal{W}$ of P, for $\varepsilon = 1/4$.

Scan all the pairs $\{u, v\}$ with a singleton as one of their sides (i.e., $|P_u| = 1$), and for each such singleton $\mathsf{P}_u = \{p\}$, record for p the closest point to it in the set P_v. Maintain for each point the closest point to it that was encountered.

We claim that in the end of this process, for every point p in P its recorded nearest point is its nearest neighbor in $\mathsf{P} \setminus \{p\}$.

Analysis. The analysis of this algorithm is slightly tedious, but it reveals some additional interesting properties of WSPD. We start with a claim that shows that we will indeed find the nearest neighbor for each point.

Lemma 20.2.7. *Let p be any point of P, and let q be the nearest neighbor to p in the set $\mathsf{P} \setminus \{p\}$. Then, there exists a pair $\{u, v\} \in \mathcal{W}$, such that $\mathsf{P}_u = \{p\}$ and $q \in \mathsf{P}_v$.*

Proof: Consider the pair $\{u, v\} \in \mathcal{W}$ such that $\{p, q\} \in \mathsf{P}_u \otimes \mathsf{P}_v$, where $p \in \mathsf{P}_u$ and $q \in \mathsf{P}_v$. Now, $\operatorname{diam}(\mathsf{P}_v) \leq \varepsilon \mathsf{d}(\mathsf{P}_u, \mathsf{P}_v) \leq \varepsilon \|pq\| \leq \|pq\|/4$. Namely, if P_v contained any other point except p, then q would not be the nearest neighbor to p. ■

Thus, the above lemma implies that the algorithm will find the nearest neighbor for each point p of P, as the appropriate pair containing only p on one side would be considered by the algorithm. Thus remains the task of bounding the running time.

A pair of nodes $\{x, y\}$ of $\mathcal{T}$ is a ***generator*** of a pair $\{u, v\} \in \mathcal{W}$ if $\{u, v\}$ was computed inside a recursive call **algWSPD**(x, y).

Lemma 20.2.8. *Let $\mathcal{W}$ be an ε^{-1}-WSPD of a point set P generated by **algWSPD**. Consider a pair $\{u, v\} \in \mathcal{W}$. Then $\Delta(\overline{\mathrm{p}}(v)) \geq (\varepsilon/2)\mathbf{d}(u, v)$ and $\Delta(\overline{\mathrm{p}}(u)) \geq (\varepsilon/2)\mathbf{d}(u, v)$, where $\mathbf{d}(u, v) = \mathsf{d}(\square_u, \square_v)$ is the distance between the cell of u and the cell of v.*

Proof: Assume, for the sake of contradiction, that $\Delta(v') < (\varepsilon/2)\ell$, where $\ell = \mathbf{d}(u, v)$ and $v' = \overline{\mathrm{p}}(v)$. By Lemma 4.1.9, we have that

$$\Delta(u) \leq \Delta(v') < \varepsilon \frac{\ell}{2}.$$

But then

$$\mathbf{d}(u, v') \geq \ell - \Delta(v') \geq \ell - \varepsilon \frac{\ell}{2} \geq \frac{\ell}{2}.$$

Thus,

$$\max(\Delta(u), \Delta(v')) < \varepsilon \frac{\ell}{2} \leq \varepsilon\, \mathbf{d}(u, v').$$

Namely, u and v' are well separated, and as such $\{u, v'\}$ cannot be a generator of $\{u, v\}$. Indeed, if $\{u, v'\}$ was considered by the algorithm, then it would have added it to the WSPD and never created the pair $\{u, v\}$.

So, the other possibility is that $\{u', v\}$ is the generator of $\{u, v\}$, where $u' = \overline{\mathrm{p}}(u)$. But then $\Delta(u') \leq \Delta(v') < \varepsilon\ell/2$, by Lemma 4.1.9. Using the same argumentation as above, we have that $\{u', v\}$ is a well-separated pair and as such it cannot be a generator of $\{u, v\}$.

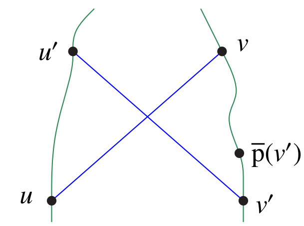
But this implies that $\{u, v\}$ cannot be generated by **algWSPD**, since either $\{u, v'\}$ or $\{u', v\}$ must be a generator of $\{u, v\}$, a contradiction. ■

Claim 20.2.9. *For two pairs $\{u, v\}, \{u', v'\} \in \mathcal{W}$ such that $\square_u \subseteq \square_{u'}$, the interiors of $\square_v$ and $\square_{v'}$ are disjoint.*

Proof: Since $\square_u \subseteq \square_{u'}$, it follows that u' is an ancestor of u.

If v' is an ancestor of v, then **algWSPD** returned the pair $\{u', v'\}$ and it would have never generated the pair $\{u, v\}$.

If v is an ancestor of v' (see the figure on the right), then



$$\begin{aligned}
\Delta(u) &< \Delta(u') && (u' \text{ is an ancestor of } u)\\
&\leq \Delta(\overline{\mathrm{p}}(v')) && (\text{by Lemma 4.1.9 applied to } \{u', v'\})\\
&\leq \Delta(v) && (v \text{ is an ancestor of } v')\\
&\leq \Delta(\overline{\mathrm{p}}(u)) && (\text{by Lemma 4.1.9 applied to } \{u, v\})\\
&\leq \Delta(u') && (u' \text{ is an ancestor of } u).
\end{aligned}$$

Namely, $\Delta(v) = \Delta(u')$. But then the pair $\{u', v\}$ is a generator of both $\{u, v\}$ and $\{u', v'\}$. To see that, observe that **algWSPD** always tries to consider pairs that have the same diameter (it always splits the bigger side of a pair). As such, for the algorithm to generate the pair $\{u, v\}$, it must have had a generator $\{u'', v\}$, such that $\operatorname{diam}(u'') \geq \operatorname{diam}(v)$. But the lowest ancestor of u that has this property is u'.

Now, when **algWSPD** considered the pair $\{u', v\}$, it split one of its sides (by calling on its children). In either case, it either did not create the pair $\{u, v\}$ or it did not create the pair $\{u', v'\}$, a contradiction.

The case $v = v'$ is handled in a similar fashion. ∎

Lemma 20.2.10. *Let P be a set of n points in $\mathbb{R}^{\mathsf{d}}$, let $\mathcal{W}$ be an ε^{-1}-WSPD of P, let $\ell > 0$ be a distance, and let L be the set of pairs $\{u, v\} \in \mathcal{W}$ such that $\ell \leq \mathbf{d}(u, v) \leq 2\ell$. Then, for any point $p \in \mathsf{P}$, the number of pairs in L containing p is $O(1/\varepsilon^{\mathsf{d}})$.*

Proof: Let u be the leaf of the quadtree $\mathcal{T}$ (that is used in computing $\mathcal{W}$) storing the point p, and let π be the path between u and the root of $\mathcal{T}$. We claim that L contains at most $O(1/\varepsilon^{\mathsf{d}})$ pairs with nodes that appear along π. Let

$$X = \left\{ v \;\middle|\; x \in \pi \text{ and } \{x, v\} \in L \right\}.$$

The cells of X are interior disjoint by Claim 4.2.9, and they contain all the pairs in W that covers p.

So, let r be the largest power of two which is smaller than (say) $\varepsilon\ell/\left(4\sqrt{\mathsf{d}}\right)$. Clearly, there are $O(1/\varepsilon^{\mathsf{d}})$ cells of G_r within a distance at most 2ℓ from $\square_u$. We account for the nodes $v \in X$, as follows:

(i) If $\Delta(v) \geq r\sqrt{\mathsf{d}}$, then $\square_v$ contains a cell of G_r, and there are at most $O(1/\varepsilon^{\mathsf{d}})$ such cells.

(ii) If $\Delta(v) < r\sqrt{\mathsf{d}}$ and $\Delta(\overline{\mathrm{p}}(v)) \geq r\sqrt{\mathsf{d}}$, then:

1. If $\overline{\mathrm{p}}(v)$ is a compressed node, then $\overline{\mathrm{p}}(v)$ contains a cell of G_r and it has only v as a single child. As such, there are at most $O(1/\varepsilon^{\mathsf{d}})$ such charges.
2. Otherwise, $\overline{\mathrm{p}}(v)$ is not compressed, but then $\operatorname{diam}(\square_v) = \operatorname{diam}\left(\square_{\overline{\mathrm{p}}(v)}\right)/2$. As such, $\square_v$ contains a cell of $\mathsf{G}_{r/2}$ within a distance at most 2ℓ from $\square_u$, and there are $O(1/\varepsilon^{\mathsf{d}})$ such cells.

(iii) The case $\Delta(\overline{\mathrm{p}}(v)) < r\sqrt{\mathsf{d}}$ is impossible. Indeed, by Lemma 4.1.9, we have $\Delta(\overline{\mathrm{p}}(v)) < r\sqrt{\mathsf{d}} \leq \varepsilon\ell/4 = (\varepsilon/4)\mathbf{d}(u, v)$, a contradiction to Lemma 4.2.8.

We conclude that there are at most $O(1/\varepsilon^{\mathsf{d}})$ pairs that include p in L. ■

Lemma 20.2.11. *Let* P *be a set of* n *points in the plane. Then one can solve the all nearest neighbors problem, in* $O(n(\log n + \log \Phi(P)))$ *time, where* Φ *is the spread of* P.

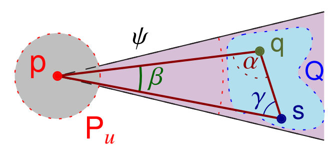
Proof: The algorithm is described above. There only remains the task of analyzing the running time. For a number $i \in \{0, -1, \ldots, -\lfloor \lg \Phi \rfloor - 4\}$, consider the set of pairs L_i, such that $\{u, v\} \in L_i$, if and only if $\{u, v\} \in \mathcal{W}$, and $2^{i-1} \leq \mathbf{d}(u, v) \leq 2^i$. Here, $\mathcal{W}$ is a $1/\varepsilon$-WSPD of P, where $\varepsilon = 1/4$. A point $p \in \mathsf{P}$ can be scanned at most $O(1/\varepsilon^{\mathsf{d}}) = O(1)$ times because of pairs in L_i by Lemma 4.2.10. As such, a point gets scanned at most $O(\log \Phi)$ times overall, which implies the running time bound. ■

20.2.5.2. All nearest neighbors – the unbounded spread case

To handle the unbounded case, we need to use some additional geometric properties.

Lemma 20.2.12. *Let* u *be a node in the compressed quadtree of* P, *and partition the space around* rep_u *into cones of angle* $\leq \pi/3$. *Let* ψ *be such a cone, and let* Q *be the set of all points in* P *which are within a distance* $\geq 4\mathrm{diam}(\mathsf{P}_u)$ *from* rep_u *and all lie inside* ψ. *Let* q *be the closest point in* Q *to* rep_u. *Then,* q *is the only point in* Q *whose nearest neighbor might be in* P_u.

Proof: Let $p = \mathrm{rep}_u$ and consider any point $\mathsf{s} \in \mathsf{Q}$.



Since $\|\mathsf{s} - p\| \geq \|q - p\|$, it follows that $\alpha = \angle \mathsf{s}qp \geq \angle q\mathsf{s}p = \gamma$. Now, $\alpha + \gamma = \pi - \beta$, where $\beta = \angle \mathsf{s}pq$. But $\beta \leq \pi/3$, and as such

$$2\alpha \geq \alpha + \gamma = \pi - \beta \geq 3\beta - \beta = 2\beta.$$

Namely, α is the largest angle in the triangle $\triangle pq\mathsf{s}$, which implies $\|\mathsf{s} - p\| \geq \|\mathsf{s} - q\|$. Namely, q is closer to s than p, and as such p cannot serve as the nearest neighbor to s in P.

It is now straightforward (but tedious) to show that, in fact, for any $t \in \mathsf{P}_u$, we have $\|\mathsf{s} - t\| \geq \|\mathsf{s} - q\|$, which implies the claim.[3] ■

Lemma 4.2.12 implies that we can do a top-down traversal of the compressed quadtree of P, after computing an ε^{-1}-WSPD $\mathcal{W}$ of P, for $\varepsilon = 1/16$. For every node u, we maintain a (constant size) set R_u of candidate points such that P_u *might* contain their nearest neighbor.

So, assume we had computed $\mathsf{R}_{\overline{\mathsf{p}}(u)}$, and consider the set

$$X(u) = \mathsf{R}_{\overline{\mathsf{p}}(u)} \cup \bigcup_{\{u,v\} \in \mathcal{W}, |\mathsf{P}_v| = 1} \mathsf{P}_v.$$

(Note that we do not have to consider pairs with $|\mathsf{P}_v| > 1$, since no point in P_v can have its nearest neighbor in P_u in such a scenario.) Clearly, we can compute $X(u)$ in linear time in the number of pairs in $\mathcal{W}$ involved with u. Now, we build a "grid" of cones around rep_u and throw the points of $X(u)$ into this grid. For each such cone, we keep only the closest point p' to rep_u (because the other points in this

[3]Here are the details for readers of little faith. By the law of sines, we have $\frac{\|p - \mathsf{s}\|}{\sin \alpha} = \frac{\|q - \mathsf{s}\|}{\sin \beta}$. As such, $\|q - \mathsf{s}\| = \|p - \mathsf{s}\| \frac{\sin \beta}{\sin \alpha}$. Now, if $\alpha \leq \pi - 3\beta$, then $\|q - \mathsf{s}\| = \|p - \mathsf{s}\| \frac{\sin \beta}{\sin \alpha} \leq \|p - \mathsf{s}\| \frac{\sin \beta}{\sin(3\beta)} \leq \frac{\|p - \mathsf{s}\|}{2} < \|p - \mathsf{s}\| - \Delta(u)$, since $\|p - \mathsf{s}\| \geq 4\Delta(u)$. This implies that no point of P_u can be the nearest neighbor of s.

If $\alpha \geq \pi - 3\beta$, then the maximum length of $q\mathsf{s}$ is achieved when $\gamma = 2\beta$. The law of sines then implies that $\|q - \mathsf{s}\| = \|p - q\| \frac{\sin \beta}{\sin(2\beta)} \leq \frac{3}{4} \|p - q\| \leq \frac{3}{4} \|p - \mathsf{s}\| < \|p - \mathsf{s}\| - \Delta(u)$, which again implies the claim.

cone would use p' as nearest neighbor before using any point of P_u). Let R_u be the set of these closest points. Since the number of cones is $O(1)$, it follows that $|\mathsf{R}_u| = O(1)$.

We continue this top-down traversal till R_u is computed for all the nodes in the tree.

Now, for every vertex u, if P_u contains only a single point p, then we compute for any point $q \in \mathsf{R}_u$ its distance to p, and if p is a better candidate to be a nearest neighbor, then we set p as the (current) nearest neighbor to q.

Correctness. Clearly, the resulting running time (ignoring the computation of the WSPD) is linear in the number of pairs of the WSPD and the size of the compressed quadtree. If p is the nearest neighbor to q, then there must be a WSPD pair $\{u, v\}$ such that $\mathsf{P}_v = \{q\}$ and $p \in \mathsf{P}_u$. But then the algorithm would add q to the set R_u, and it would be in R_z, for all descendants z of u in the quadtree, such that $p \in \mathsf{P}_z$. In particular, if y is the leaf of the quadtree storing p, then $q \in \mathsf{R}_y$, which implies that the algorithm computes correctly the nearest neighbor to q.

This implies the correctness of the algorithm.

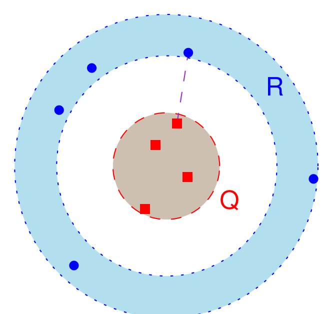
Theorem 20.2.13. *Given a set P of n points in $\mathbb{R}^{\mathsf{d}}$, one can solve the all nearest neighbors problem in $O(n \log n)$ time.*

20.3. Semi-separated pair decomposition (SSPD)

Here we present an interesting relaxation of WSPD that has the advantage of having a low total weight.

Definition 20.3.1. Given a pair decomposition $\mathcal{W} = \Big\{\{A_1, B_1\}, \ldots, \{A_s, B_s\}\Big\}$ of a point set P, its ***weight*** is $\omega(\mathcal{W}) = \sum_{i=1}^{s}(|A_i| + |B_i|)$.

It is easy to verify that, in the worst case, a WSPD of a set of n points in the plane might have weight $O(n^2)$; see Exercise 4.5.1. The notation of SSPD circumvents this by requiring a weaker notion of separation.



Definition 20.3.2. Two sets of points Q and R are ***$(1/\varepsilon)$-semi-separated*** if

$$\min(\operatorname{diam}(\mathsf{Q}), \operatorname{diam}(\mathsf{R})) \leq \varepsilon \cdot \mathsf{d}(\mathsf{Q}, \mathsf{R}),$$

where $\mathsf{d}(\mathsf{Q}, \mathsf{R}) = \min_{\mathsf{s} \in \mathsf{Q}, t \in \mathsf{R}} \|\mathsf{s}t\|$.

See the figure on the right for an example.

See Definition 4.1.2 for the original notion of sets being well separated; in particular, in the well-separated case we demanded that the separation be large relative to the diameter of both sets, while here it is sufficient that the separation be large for the smaller of the two sets. The following is the analog of the WSPD (Definition 4.1.3).

Definition 20.3.3 (SSPD). For a point set P, a ***semi-separated pair decomposition*** (***SSPD***) of P with parameter $1/\varepsilon$, denoted by ε^{-1}-SSPD, is a pair decomposition of P formed by a set of pairs $\mathcal{W}$ such that all the pairs are $1/\varepsilon$-semi-separated.

Observation 20.3.4. *An ε^{-1}-WSPD of P is an ε^{-1}-SSPD of P.*

20.3.1. Construction

We use the property that in low dimensions there always exists a good separating ring that breaks the point set into two reasonably large subsets. Indeed, consider the smallest ball $\mathsf{b} = \mathsf{b}(p, r)$ that contains n/c_1 points of P, where c_1 is a sufficiently large constant. Let b' be the scaling of this ball by a factor of two. By a standard packing argument, the ring $\mathsf{b}' \setminus \mathsf{b}$ can be covered with $\mathsf{c} = O(1)$ copies of b, none of which can contain more than n/c_1 points of P, see Figure 4.3.1. It follows that by picking $\mathsf{c}_1 = 3\mathsf{c}$, we are guaranteed that at least half the points of P are outside b'. Now, the ring can be split into $n/2$ empty rings (by taking a sphere that passes through each point inside the ring). One of them would be of thickness at least r/n, and it would separate the inner n/c points of P from the outer $n/2$ points of P. Doing this efficiently requires trading off some constants, and it requires some tedious details, as described in the following lemma.

Lemma 20.3.5. *Let P be a set of n points in $\mathbb{R}^{\mathsf{d}}$, let $t > 0$ be a parameter, and let c be a sufficiently large constant. Then one can compute, in linear time, a ball $\mathsf{b} = \mathsf{b}(p, r)$, such that*

(i) $|\mathsf{b} \cap \mathsf{P}| \geq n/c$,

(ii) $|\mathsf{b}(p, r(1 + 1/t)) \cap \mathsf{P}| \leq n/2t + |\mathsf{b} \cap \mathsf{P}|$, *and*

(iii) $|\mathsf{P} \setminus \mathsf{b}(p, 2r)| \geq n/2$.

Proof: Let $\mathsf{b} = \mathsf{b}(p, \alpha)$ be the disk computed, in $O(n)$ time, by Lemma 4.6.1, for $k = n/c$. This ball contains n/c points of P and $\alpha \leq 2r_{\mathrm{opt}}(\mathsf{P}, k)$, where $r_{\mathrm{opt}}(\mathsf{P}, k)$ is the radius of the smallest ball containing k points of P. Observe that the ball $\mathsf{b}_{8\alpha} = \mathsf{b}(p, 8\alpha)$ can be covered by $M = O(1)$ balls of radius $\alpha/2$. Each of these balls contains at most n/c points, by the construction of b. As such, if $c > 2M$, we have that $|\mathsf{P} \cap \mathsf{b}_{8\alpha}| \leq M(n/c) \leq n/2$ and $|\mathsf{P} \setminus \mathsf{b}_{8\alpha}| \geq n/2$.

We will set $r \in [\alpha, e\alpha]$ in such a way that property (ii) will hold for it. Indeed, set $r_i = \alpha(1 + 1/t)^i$, for $i = 0, \ldots, t$, and consider the rings

$$\mathcal{R}_i = \mathsf{b}(p, r_i) \ \setminus \ \mathsf{b}(p, r_{i-1}),$$

for $i = 1, \ldots, t$. We have that $r_t = \alpha(1 + 1/t)^t \leq \alpha \exp(t/t) = \alpha e$, since $1 + x \leq e^x$ for all $x \geq 0$. Now, all these (interior disjoint) rings are contained inside $\mathsf{b}_{4\alpha}$. It follows that one of these rings, say the ith ring $\mathcal{R}_i$, contains at most $(n/2)/t$ of the points of P (since $\mathsf{b}(p, 8\alpha)$ contains at most half of the points of P). For $r = r_{i-1} \leq 4r$ the ball $\mathsf{b} = \mathsf{b}(p, r)$ has the required properties, as $\mathsf{b}(p, 2r) \subseteq \mathsf{b}(p, 8\alpha)$. ■

We also need the following easy property.

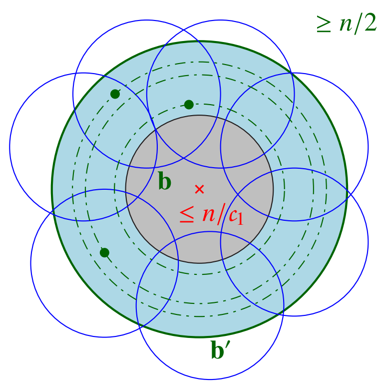


Figure 20.3.1: Cover ring.

Lemma 20.3.6. *Let* P *be a set of* n *points in* $\mathbb{R}^{\mathsf{d}}$*, with spread* $\Phi = \Phi(\mathsf{P})$*, and let* $\varepsilon > 0$ *be a parameter. Then, one can compute* $(1/\varepsilon)$*-WSPD (and thus a* $(1/\varepsilon)$*-SSPD) for* P *of total weight* $O(n\varepsilon^{-\mathsf{d}} \log \Phi)$*. Furthermore, any point of* P *participates in at most* $O\Big(\varepsilon^{-\mathsf{d}} \log \Phi\Big)$ *pairs.*

Proof: Build a regular (i.e., not compressed) quadtree for P, and observe that its depth is $O(\log \Phi)$. Now, construct a WSPD for P using this quadtree. Consider a pair of nodes (u, v) in this WSPD, and observe that the sidelength of u and v is the same up to a factor of two (since we used a non-compressed quadtree). As such, every node participates in $O(1/\varepsilon^{\mathsf{d}})$ pairs in the WSPD. We conclude that each point participates in $O\Big(\varepsilon^{-\mathsf{d}} \log \Phi\Big)$ pairs, which implies that the total weight of this WSPD is as claimed. ■

Theorem 20.3.7. *Let* P *be a set of points in* $\mathbb{R}^{\mathsf{d}}$*, and let* $\varepsilon > 0$ *be a parameter. Then, one can compute a* ε^{-1}*-SSPD for* P *of total weight* $O\Big(n\varepsilon^{-\mathsf{d}} \log^2 n\Big)$*. The number of pairs in the SSPD is* $O\Big(n\varepsilon^{-\mathsf{d}} \log n\Big)$*, and the computation time is* $O\Big(n \log^2 n + n\varepsilon^{-\mathsf{d}} \log n\Big)$*.*

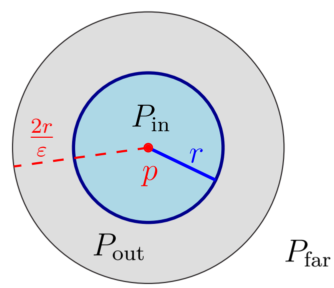
Proof: Using Lemma 4.3.5, with $t = n$, we compute a ball $\mathsf{b}(p, r)$ that contains at least n/c points of P and such that $\mathcal{R} = \mathsf{b}(p, (1 + 1/t)r) \setminus \mathsf{b}(p, r)$ contains no point of P.

Let $\mathsf{P}_{\text{in}} = \mathsf{P} \cap \mathsf{b}$,

$$\mathsf{P}_{\text{out}} = (\mathsf{P} \setminus \mathsf{P}_{\text{in}}) \cap \mathsf{b}(p, 2r/\varepsilon),$$

and

$$\mathsf{P}_{\text{far}} = \mathsf{P} \setminus \Big(\mathsf{P}_{\text{in}} \cup \mathsf{P}_{\text{out}}\Big).$$



Clearly, $\{\mathsf{P}_{\text{in}}, \mathsf{P}_{\text{far}}\}$ is a $(1/\varepsilon)$-semi-separated pair, which we add to our SSPD. Let $\ell = \min_{p \in \mathsf{P}_{\text{in}}, q \in \mathsf{P}_{\text{out}}} \|pq\|$. Observe that ℓ is larger than the thickness of the empty ring $\mathcal{R}$; that is, $\ell \geq r/n$.

We would like to compute the SSPD for all pairs in $\mathsf{P}_{\text{in}} \otimes \mathsf{P}_{\text{out}}$. The observation is that none of these pairs are of distance smaller than ℓ, and the diameter of the point set $\mathsf{Q} = \mathsf{P}_{\text{in}} \cup \mathsf{P}_{\text{out}}$ is $\text{diam}(\mathsf{Q}) \leq 4\ell n/\varepsilon$. Thus, we can snap the point set Q to a grid of sidelength $\varepsilon\ell/(10\mathsf{d})$. The resulting point set Q' has spread $O(n/\varepsilon^2)$. Next, compute a $2/\varepsilon$-SSPD for the snapped point set Q', using Lemma 4.3.6. Clearly, the computed SSPD when extended back to the point set Q would cover all the pairs of $\mathsf{P}_{\text{in}} \otimes \mathsf{P}_{\text{out}}$, and it would provide a $(1/\varepsilon)$-SSPD for these pairs. By Lemma 4.3.6, every point of Q would participate in at most $O\Big(\varepsilon^{-\mathsf{d}} \log(n/\varepsilon)\Big) = O\Big(\varepsilon^{-\mathsf{d}} \log n\Big)$ pairs.

To complete the construction, we need to construct a $(1/\varepsilon)$-SSPD for the pairs in $\mathsf{P}_{\text{in}} \otimes \mathsf{P}_{\text{in}}$ and in $(\mathsf{P}_{\text{out}} \cup \mathsf{P}_{\text{far}}) \otimes (\mathsf{P}_{\text{out}} \cup \mathsf{P}_{\text{far}})$. This we do by continuing the construction recursively on the point sets P_{in} and $\mathsf{P}_{\text{out}} \cup \mathsf{P}_{\text{far}}$.

In the resulting WSPD, every point participates in at most

$$T(n) = 1 + O\Big(\varepsilon^{-\mathsf{d}} \log n\Big) + \max\Big(T(n_1), T(n_2)\Big)$$

pairs of the resulting SSPD, where $n_1 = |\mathsf{P}_{\text{in}}|$ and $n_2 = |\mathsf{P}_{\text{out}} \cup \mathsf{P}_{\text{far}}|$. Since $n_1 + n_2 = n$ and $n_1, n_2 \geq n/c$, where c is some constant, it follows that $T(n) = O\Big(\varepsilon^{-\mathsf{d}} \log^2 n\Big)$. Namely, every point of P participates in at most $O\Big(\varepsilon^{-\mathsf{d}} \log^2 n\Big)$ pairs in the resulting SSPD. As such, the total weight of the SSPD is $O\Big(n\varepsilon^{-\mathsf{d}} \log^2 n\Big)$.

In each level of the recursion, we create at most $O(n/\varepsilon^2)$ pairs. As such, the bound on the number of pairs created follows. The bound on the running time follows by similar argumentation. ■

20.3.2. Lower bound

The result of Theorem 4.3.7 can be improved so that the total weight of the SSPD is $O\left(n\varepsilon^{-\mathsf{d}}\log n\right)$; see the bibliographical notes in the next section. Interestingly, it turns out that any pair decomposition (without any required separation property) has to be of total weight $\Omega(n\log n)$.

Lemma 20.3.8. *Let P be a set of n points, and let $\mathcal{W} = \left\{\{A_1, B_1\}, \ldots, \{A_s, B_s\}\right\}$ be a pair decomposition of P (see Definition 4.1.1). Then, $\omega(\mathcal{W}) = \sum_i (|A_i| + |B_i|) = \Omega(n\log n)$.*

Proof: Scan the pairs $\{A_i, B_i\}$ one by one, and for each such pair, randomly set Y_i to be either A_i or B_i with equal probability. Let $\mathsf{R} = \mathsf{P} \setminus \left(\bigcup_i Y_i\right)$.

Observe that R is either empty or contain a single point. Indeed, if there are two distinct points $p, q \in \mathsf{R}$, then there exists an index j such that $\{p,q\} \in A_j \otimes B_j$. In particular, $|\{p,q\} \cap A_j| = 1$ and $|\{p,q\} \cap B_j| = 1$. As such $|\{p,q\} \cap Y_j| = 1$, and this implies that R cannot contain both points, as R does not contain any of the points of Y_j.

Now, let $\mathsf{P} = \{p_1, \ldots, p_n\}$ and let x_i be the number of pairs that contain p_i (in either side of the pair), for $i = 1, \ldots, n$. Now, the quantity we need to bound is $\sum_i x_i$ since $\sum_{i=1}^s (|A_i| + |B_i|) = \sum_{i=1}^n x_i$. Observe that the probability of p_i to be in R is *exactly* $1/2^{x_i}$ since this is the probability that for all the x_i pairs that contain p_i we eliminated the side that does not contain p_i. As such, define an indicator variable Z_i such that $Z_i = 1$ if and only if $p_i \in \mathsf{R}$. Observe that $\sum_i Z_i = |\mathsf{R}| \leq 1$. As such, by linearity of expectations, we have that $\sum_i \mathbb{E}[Z_i] = \mathbb{E}[\sum_i Z_i] \leq 1$. But $\mathbb{E}[Z_i = 1] = \mathbb{P}[p_i \in \mathsf{R}] = 1/2^{x_i}$. Namely, we have that

$$\sum_{i=1}^n \frac{1}{2^{x_i}} = \sum_i \mathbb{P}[p_i \in \mathsf{R}] = \sum_i \mathbb{E}[Z_i] \leq 1.$$

Let $u_1, \ldots, u_n$ be a set of n positive integers that minimizes $\sum_i u_i$, such that $\sum_{i=1}^n 2^{-u_i} \leq 1$. Observe that $\sum_i u_i \leq \sum_i x_i$.

Now, if there are i and j such that $u_i > u_j + 1$, then we have that

$$\frac{1}{2^{u_i - 1}} + \frac{1}{2^{u_j+1}} \leq \frac{2}{2^{u_j+1}} = \frac{1}{2^{u_j}} \leq \frac{1}{2^{u_i}} + \frac{1}{2^{u_j}}.$$

Namely, setting $u_i - 1$ and $u_j + 1$ as the new values of u_i and u_j, respectively, does not change the quantity $u_1 + \cdots + u_n$, while preserving the inequality $\sum_{i=1}^n 2^{-u_i} \leq 1$. We repeat this fix-up process till we have $|u_i - u_j| \leq 1$ for all i and j. So, let t be the number such that $t \leq u_i \leq t+1$, for all i. We have that

$$\sum_{i=1}^n 1/2^{t+1} \leq \sum_{i=1}^n 1/2^{u_i} \leq 1 \implies n/2^{t+1} \leq 1 \implies n \leq 2^{t+1} \implies t \geq \lg(n/2).$$

We conclude that $\sum_i x_i \geq \sum_i u_i \geq nt \geq n\lg(n/2)$, as claimed. ■

20.4. Bibliographical notes

Well-separated pair decomposition was defined by Callahan and Kosaraju [CK95]. They defined a different space decomposition tree, known as the ***fair split tree***. Here, one computes the axis parallel bounding box of the point set, always splitting along the longest edge by a perpendicular plane in the middle (or near the middle). This splits the point set into two sets, for which we construct the fair split tree recursively. Implementing this in $O(n\log n)$ time requires some cleverness. See [CK95] for details.

Our presentation of the WSPD (very roughly) follows [HM06]. The (easy) observation that a WSPD can be generated directly from a compressed quadtree (thus avoiding the fair split tree) is from there.

Callahan and Kosaraju [CK95] were inspired by the work of Vaidya [Vai86] on the all nearest neighbors problem (i.e., compute for each point in P its nearest neighbor in P). He defined the fair split tree and showed how to compute the all nearest neighbors in $O(n \log n)$ time. However, the first to give an $O(n \log n)$ time algorithm for the all nearest neighbors problem was Clarkson [Cla83] using similar ideas (this was part of his PhD thesis).

Diameter. The algorithm for computing the diameter in Section 4.2.3 can be improved by not constructing pairs that cannot improve the (current) diameter and by constructing the underlying tree on the fly together with the diameter. This yields a simple algorithm that works quite well in practice; see [Har01].

All nearest neighbors.

Section 4.2.5 is a simplification of the algorithm for the all k-nearest neighbors problem. Here, one can compute for every point its k-nearest neighbors in $O(n \log n + nk)$ time. See [CK95] for details.

The all nearest neighbors algorithm for the bounded spread case (Section 4.2.5.1) is from [HM06]. Note that unlike the unbounded case, this algorithm only uses packing arguments for its correctness. Surprisingly, the usage of the Euclidean nature of the underlying space (as done in Section 4.2.5.2) seems to be crucial in getting a faster algorithm for this problem. In particular, for the case of metric spaces of low doubling dimension (that do have a small WSPD), solving this problem requires $\Omega(n^2)$ time in the worst case.

Note, that the all nearest neighbors graph is a subgraph of the MST, and more importantly for our purposes, it is also a subgraph of the Delaunay triangulation. As such, in the plane, it can be computed in $O(n \log n)$ time. However, in higher dimensions, no exact MST algorithm is known with near linear running time.

Dynamic maintenance. WSPD can be maintained in polylogarithmic time under insertions and deletions. This is quite surprising when one considers that, in the worst case, a point might participate in a linear number of pairs, and a node in the quadtree might participate in a linear number of pairs. This is described in detail in Callahan's thesis [Cal95]. Interestingly, using randomization, maintaining the WSPD can be considerably simplified; see the work by Fischer and Har-Peled [FH05].

High dimension. In high dimensions, as the uniform metric demonstrates (i.e., n points, all of them within a distance of 1 from each other), the WSPD can have quadratic complexity. This metric is easily realizable as the vertices of a simplex in $\mathbb{R}^{n-1}$. On the other hand, doubling metrics have near linear size WSPD. Since WSPDs by themselves are so powerful, it is tempting to try to define the dimension of a point set by the size of the WSPD it has. This seems like an interesting direction for future research, as currently little is known about it (to the best of the author's knowledge).

Semi-separated pair decomposition. The notion of semi-separated pair decomposition was introduced by Varadarajan [Var98] who used it to speed up the matching algorithm for points in the plane. His SSPD construction was of total weight $O(n \log^4 n)$ (for a constant ε). This was improved to $O(n \log n)$ by [ABFG09].

SSPDs were used to construct spanners that can survive even if a large fraction of the graph disappears (for example, all the nodes inside the arbitrary convex region disappear) [ABFG09]. In [ABF+11], SSPDs were used for computing additively weighted spanners. Further work on SSPDs can be found in [ACFS09]. The simpler construction shown here is due to Abam and Har-Peled [AH10] and it also works for metrics with low doubling dimension.

The elegant proof of the lower bound on the size of the SSPD is from [BS07].

Euclidean minimum spanning tree. Let P be a set of points in $\mathbb{R}^d$ for which we want to compute its minimum spanning tree (MST). It is easy to verify that if an edge pq is not a Delaunay edge of P, then its diametrical ball must contain some other point s in its interior, but then $p\mathsf{s}$ and $q\mathsf{s}$ are shorter than pq. This in turn implies that pq cannot be an edge of the MST. As such, the edges of the MST are subset of the edges of the Delaunay triangulation of P. Since the Delaunay triangulation can be computed in $O(n \log n)$ time in the plane, this implies that the MST can be computed in $O(n \log n)$ time in the plane. In $\mathsf{d} \geq 3$ dimensions, the exact MST can be computed in $O\Big(n^{2-2/(\lceil \mathsf{d}/2 \rceil+1)+\varepsilon'}\Big)$ time, where $\varepsilon' > 0$ is an arbitrary small constant [AESW91]. The computation of the MST is closely related to the bichromatic closest pair problem [KLN99].

There is a lot of work on the MST in Euclidean settings, from estimating its weight in sublinear time [CRT05], to doing this in the streaming model under insertions and deletions [FIS05], to implementation of a variant of the approximation algorithm described here [NZ01]. This list is by no means exhaustive.

20.5. Exercises

Exercise 20.5.1 (The unbearable weight of the WSPD). Show that there exists a set of n points (even on the line), such that any ε^{-1}-WSPD for it has weight $\Omega(n^2)$, for $\varepsilon = 1/4$.

Exercise 20.5.2 (WSPD structure). Let $\varepsilon > 0$ be a sufficiently small constant. For any n sufficiently large, show an example of a point set P of n points, such that its $(1/\varepsilon)$-WSPD (as computed by **algWSPD**) has the property that a single set participates in $\Omega(n)$ sets.[4]

Exercise 20.5.3 (Number of resolutions that matter). Let P be an n-point set in $\mathbb{R}^{\mathsf{d}}$, and consider the set

$$U = \left\{ i \;\middle|\; 2^i \leq \|p - q\| \leq 2^{i+1}, \text{ for } p, q \in \mathsf{P} \right\}.$$

Prove that $|U| = O(n)$ (the constant depends on d). Namely, there are only n different resolutions that "matter".

Exercise 20.5.4 (WSPD and sum of distances). Let P be a set of n points in $\mathbb{R}^{\mathsf{d}}$. The ***sponginess***[5] of P is $X = \sum_{\{p,q\} \subseteq \mathsf{P}} \|p - q\|$. Provide an efficient algorithm for approximating X. Namely, given P and a parameter $\varepsilon > 0$, it outputs a number Y such that $X \leq Y \leq (1 + \varepsilon)X$.

(The interested reader can also verify that computing (exactly) the sum of all ***squared*** distances (i.e., $\sum_{\{p,q\} \subseteq \mathsf{P}} \|p - q\|^2$) is considerably easier.)

Exercise 20.5.5 (SSPD with fewer pairs). Let P be a set of n points in $\mathbb{R}^{\mathsf{d}}$. The result of Theorem 4.3.7 can be improved so that the number of pairs in the $(1/\varepsilon)$-SSPD $\mathcal{W}$ generated is $O(n/\varepsilon^{\mathsf{d}})$. To this end, consider a pair separating points of P_{in} from P_{far} as a ***long pair***, and a pair separating P_{in} from P_{out} as a ***short pair***.

(A) Prove that the number of long pairs generated by the construction of Theorem 4.3.7 is $O(n)$.
(B) For an appropriately small enough constant c, consider a (c/ε)-WSPD $\mathcal{W}'$ of P, and show how to find for each short pair of $\mathcal{W}$ the pair in $\mathcal{W}'$ that looks like it.
(C) Show how to reduce the number of short pairs in the SSPD by merging certain pairs, so that the resulting pairs are still $O(1/\varepsilon)$-separated.
(D) Describe how to reduce the number of pairs $\mathcal{W}$, so that the resulting decomposition is an $O(1/\varepsilon)$-SSPD with $O(n/\varepsilon^{\mathsf{d}})$ pairs and having the same weight as $\mathcal{W}$.

[4] Note that there is always a WSPD construction such that each node participates in a "small" number of pairs.
[5] This is also known as the sum of pairwise distances in the literature, for reasons that the author cannot fathom.

20.6. From previous lectures

Lemma 20.6.1. *Given a set* P *of* n *points in* $\mathbb{R}^{\mathsf{d}}$ *and parameter* k*, one can compute, in* $O\Big(n(n/k)^{\mathsf{d}}\Big)$ *deterministic time, a ball* b *that contains* k *points of* P *and its radius* $\mathrm{radius}(\mathsf{b}) \leq 2r_{\mathrm{opt}}(\mathsf{P}, k)$*, where* $r_{\mathrm{opt}}(\mathsf{P}, k)$ *is the radius of the smallest ball in* $\mathbb{R}^{\mathsf{d}}$ *containing* k *points of* P*.*

Definition 20.6.2 (Canonical squares and canonical grids). A square is a ***canonical square*** if it is contained inside the unit square, it is a cell in a grid G_r, and r is a power of two (i.e., it might correspond to a node in a quadtree). We will refer to such a grid G_r as a ***canonical grid***.

References

[ABF+11] M. A. Abam, M. de Berg, M. Farshi, J. Gudmundsson, and M. H. M. Smid. Geometric spanners for weighted point sets. *Algorithmica*, 61(1): 207–225, 2011.

[ABFG09] M. A. Abam, M. de Berg, M. Farshi, and J. Gudmundsson. Region-fault tolerant geometric spanners. *Discrete Comput. Geom.*, 41(4): 556–582, 2009.

[ACFS09] M. A. Abam, P. Carmi, M. Farshi, and M. H. M. Smid. On the power of the semi-separated pair decomposition. *Proc. 11th Workshop Algorithms Data Struct.* (WADS), 1–12, 2009.

[AESW91] P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and E. Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete Comput. Geom.*, 6(5): 407–422, 1991.

[AH10] M. A. Abam and S. Har-Peled. New constructions of SSPDs and their applications. *Proc. 26th Annu. Sympos. Comput. Geom.* (SoCG), 192–200, 2010.

[BS07] B. Bollobás and A. Scott. On separating systems. *Eur. J. Comb.*, 28(4): 1068–1071, 2007.

[Cal95] P. B. Callahan. *Dealing with higher dimensions: the well-separated pair decomposition and its applications.* Ph.D. Thesis. Baltimore, Maryland: Dept. Comput. Sci., Johns Hopkins University, 1995.

[CK95] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. Assoc. Comput. Mach.*, 42(1): 67–90, 1995.

[Cla83] K. L. Clarkson. Fast algorithms for the all nearest neighbors problem. *Proc. 24th Annu. IEEE Sympos. Found. Comput. Sci.* (FOCS), 226–232, 1983.

[CRT05] B. Chazelle, R. Rubinfeld, and L. Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM J. Comput.*, 34(6): 1370–1379, 2005.

[FH05] J. Fischer and S. Har-Peled. Dynamic well-separated pair decomposition made easy. *Proc. 17th Canad. Conf. Comput. Geom.* (CCCG), 235–238, 2005.

[FIS05] G. Frahling, P. Indyk, and C. Sohler. Sampling in dynamic data streams and applications. *Proc. 21st Annu. Sympos. Comput. Geom.* (SoCG), 142–149, 2005.

[Har01] S. Har-Peled. A practical approach for computing the diameter of a point-set. *Proc. 17th Annu. Sympos. Comput. Geom.* (SoCG), 177–186, 2001.

[HM06] S. Har-Peled and M. Mendel. Fast construction of nets in low dimensional metrics, and their applications. *SIAM J. Comput.*, 35(5): 1148–1184, 2006.

[KLN99] D. Krznaric, C. Levcopoulos, and B. J. Nilsson. Minimum spanning trees in d dimensions. *Nordic J. Comput.*, 6(4): 446–461, 1999.

[NZ01] G. Narasimhan and M. Zachariasen. Geometric minimum spanning trees via well-separated pair decompositions. *J. Exp. Algorithmics*, 6: 6, 2001.

[Vai86] P. M. Vaidya. An optimal algorithm for the all-nearest-neighbors problem. *Proc. 27th Annu. IEEE Sympos. Found. Comput. Sci.* (FOCS), 117–122, 1986.

[Var98] K. R. Varadarajan. A divide-and-conquer algorithm for min-cost perfect matching in the plane. *Proc. 39th Annu. IEEE Sympos. Found. Comput. Sci.* (FOCS), 320–331, 1998.