Chapter 3

Well Separated Pairs Decomposition

By Sariel Har-Peled, January 16, 2008[①]

In this chapter, we will investigate of 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.

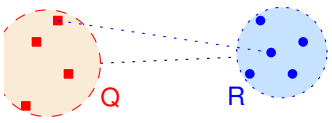
3.1 Well-separated pairs decomposition

Let P be a set of n points in $\mathbb{R}^d$, and $1/4 > \varepsilon > 0$ a parameter. One can represent the 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 dn), but its not a very informative representation. We interested in a representation that would 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 that p has similar distance to q and r, and furthermore, the q and r 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 r, since they are the only pair (among the three) that might realize the closest pair.

Figure 3.1

Denote by $A \otimes B = \left\{ \{x, y\} \,\middle|\, x \in A,\ y \in B \right\}$ all the (unordered) pair of points formed by the sets A and B.. A pair of sets of points Q and R is $(1/\varepsilon)$-***separated*** if



$$\max(\mathrm{diam}(\mathsf{Q}), \mathrm{diam}(\mathsf{R})) \leq \varepsilon \cdot \mathbf{d}(\mathsf{Q}, \mathsf{R}),$$

where $\mathbf{d}(\mathsf{Q}, \mathsf{R}) = \min_{\mathsf{q} \in \mathsf{Q}, \mathsf{r} \in \mathsf{R}} \|\mathsf{q} - \mathsf{r}\|$. 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 now we require that the distance between the two balls is at least $2/\varepsilon$ the radius of the larger of the two.

Thus, for the three points of Figure 3.1, the pairs $\{\mathsf{p}\} \otimes \{\mathsf{q}, \mathsf{r}\}$ and $\{\mathsf{q}\} \otimes \{\mathsf{r}\}$ are (say) 2-separated and described 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 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 3.1.1 (WSPD) A ***well-separated pair decomposition*** (***WSPD***) with parameter $1/\varepsilon$ of P is a set of pairs

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

such that (A) $A_i, B_i \subset \mathsf{P}$ for every i.

(B) $A_i \cap B_i = \emptyset$ for every i.

(C) $\cup_{i=1}^{s} A_i \otimes B_i = \mathsf{P} \otimes \mathsf{P}$.

(D) The sets A_i and B_i are ε^{-1}-separated.

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

For a concrete example of a WSPD, see Figure 3.2.

Instead of maintaining such a decomposition explicitly, it is convenient to construct a tree $\mathcal{T}$ having the points of P as leaves, and 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 denote the points of P stored in the subtree of v, where 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.

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

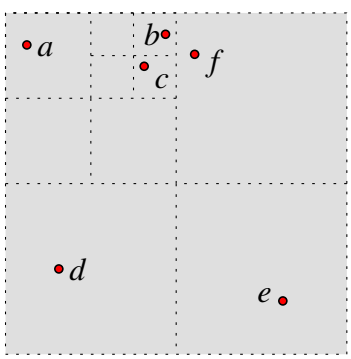
Corollary 3.1.2 *For a ε^{-1}-WSPD $\mathcal{W}$, it holds, for any pair $\{u, v\} \in \mathcal{W}$, that*

$$\forall \mathsf{q} \in \mathsf{P}_u, \mathsf{r} \in \mathsf{P}_v \qquad \max(\operatorname{diam}(\mathsf{P}_u), \operatorname{diam}(\mathsf{P}_v)) \leq \varepsilon \|\mathsf{q} - \mathsf{r}\|.$$

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 $\mathcal{T}$ used to represent the WSPD.

3.1.1 The construction algorithm

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 than the pair formed by the parents of this pair of nodes will not be well separated. As such, the algorithm starts from the



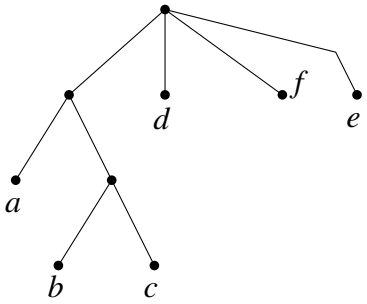
(i)

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

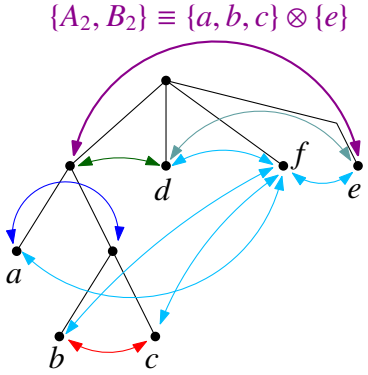
(ii)

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

(iii)



(iv)



(v)

Figure 3.2: (i) A point set $\mathsf{P} = \{a, b, c, d, e\}$, (i) its decomposition into pairs, and (iii) its 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 $\mathcal{T}$ representing the point set P. (v) The WSPD as defined by pairs of vertices of $\mathcal{T}$.

root, and try 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.

Let $\Delta(v)$ denote the diameter of a cell associated with a node v of the quadtree $\mathcal{T}$. 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 $\Delta(v) = \operatorname{diam}(\square_v)$, where $\square_v$ (we remind the reader) is the quadtree cube 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

$$\mathbf{d}(u, v) = \mathbf{d}(\square_u, \square_v) = \min_{\mathsf{p}\in\square_u, \mathsf{q}\in\square_v} \|\mathsf{p} - \mathsf{q}\|.$$

AlgWSPD(u, v, T)
  if $\Delta(u) < \Delta(v)$ **then**
    Exchange u and v
  If $\Delta(u) \le \varepsilon \cdot \mathbf{d}(u, v)$ **then**
    return $\{\{u, v\}\}$

  // $u_1, \ldots, u_r$ - the children of u
  return $\bigcup_{i=1}^{r}$ **AlgWSPD**(u_i, v, T).

Figure 3.3: The algorithm **AlgWSPD** for computing well-separated pairs decomposition. The nodes u and v belong to a compressed quadtree $\mathcal{T}$ of P.

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), where u_0 is the root of $\mathcal{T}$ and **AlgWSPD** is depicted in Figure 3.3.

The following lemma is implied by an easy packing argument.

Lemma 3.1.3 *Let $\square$ be a cell of a grid G of $\mathbb{R}^d$ with cell diameter x. For $y \ge x$, the number of cells in G at distance at most y from $\square$ is $O\big((y/x)^d\big)$.*[2]

Lemma 3.1.4 *The WSPD generated by **AlgWSPD** is valid. Namely, for any pair $\{u, v\}$ in the WSPD, we have*

$$\max(\operatorname{diam}(\mathsf{P}_u), \operatorname{diam}(\mathsf{P}_v)) \le \varepsilon \cdot \mathbf{d}(u, v) \quad \text{and} \quad \mathbf{d}(u, v) \le \|\mathsf{q} - \mathsf{r}\|,$$

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

Proof: For every output pair $\{u, v\}$, we have

$$\max\{\operatorname{diam}(\mathsf{P}_u), \operatorname{diam}(\mathsf{P}_v)\} \quad \le \quad \max\{\Delta(u), \Delta(v)\} \le \frac{\varepsilon}{8}\mathbf{d}(u, v) \le \varepsilon \cdot \mathbf{d}(u, v).$$

Also, for any $\mathsf{q} \in \mathsf{P}_u$ and $\mathsf{r} \in \mathsf{P}_v$, we have

$$\mathbf{d}(u, v) = \mathbf{d}(\square_u, \square_v) \le \mathbf{d}(\mathsf{P}_u, \mathsf{P}_v) \le \|\mathsf{q} - \mathsf{r}\|,$$

since $\mathsf{P}_u \subseteq \square_u$ and $\mathsf{P}_v \subseteq \square_v$.

Finally, 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. ∎

[2] The $O(\cdot)$ notation here (and the rest of the chapter) hides a constant that depends on d.

Lemma 3.1.5 *For a pair $\{u, v\} \in \mathcal{W}$ computed by* **AlgWSPD**, *we have that*

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

Proof: We trivially have that $\Delta(u) < \Delta(\overline{\mathrm{p}}(u))$ and $\Delta(v) < \Delta(\overline{\mathrm{p}}(v))$.

The pair $\{u, v\}$ was 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 refine the larger cell (i.e., $v_{s-1} = \overline{\mathrm{p}}(v)$ in the pair $\{u_{s-1}, v_{s-1}\}$).

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

Lemma 3.1.6 *The number of pairs in the computed* WSPD *is* $O(n/\varepsilon^d)$.

Proof: Let $\{u, v\}$ be an output pair. 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 3.1.5.

We charge the pair $\{u, v\}$ to the node v', and claim that each node of $\mathcal{T}$ is charged at most $O(\varepsilon^{-d})$ times. To this end, fix a node $v' \in V(T)$, where $V(T)$ is the set of vertices of $\mathcal{T}$. Since the pair $\{u, v'\}$ was not output by **AlgWSPD** (despite being considered) we conclude that $8\Delta(v') > \varepsilon \cdot \mathbf{d}(u, v')$ and as such $\mathbf{d}(u, v') < r = 8\Delta(v')/\varepsilon$. Now, there are several possibilities:

(i) $\Delta(v') = \Delta(u)$. But there are at most $O\Big((r/\Delta(v'))^d\Big) = O(1/\varepsilon^d)$ nodes that have the same level (i.e., diameter) as v' and their cells are in distance at most r from it, by Lemma 3.1.3. Thus, this type of charge can happened at most $O(2^d \cdot (1/\varepsilon^d))$ 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 $O(1/\varepsilon^d)$ 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 $O\Big(2^d \cdot 2^d \cdot (1/\varepsilon^d)\Big)$.

(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 3.5.1). 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, $\mathbf{d}(\widehat{\square}, \square_{v'}) \leq \mathbf{d}(\square_u, \square_{v'}) = \mathbf{d}(u, v') < r$. It follows that are at most $O(1/\varepsilon^d)$ cells like $\widehat{\square}$ that might participate in charging v', and as such, the total number of charges is $O(2^d/\varepsilon^d)$, as claimed.

As such, v' can be charged at most $O\Big(2^{2d}/\varepsilon^d\Big) = O\Big(1/\varepsilon^d\Big)$ times. This implies that the total number of pairs generated by the algorithm is $O(n\varepsilon^{-d})$, 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 3.1.7 *For $1 \geq \varepsilon > 0$, one can construct a ε^{-1}-WSPD of size $n\varepsilon^{-d}$, and the construction time is $O\left(n \log n + n\varepsilon^{-d}\right)$. Furthermore, for any pair $\{u, v\}$ in the WSPD, we have*

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

3.2 Applications of WSPD

3.2.1 Spanners

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

$$\|\mathsf{q} - \mathsf{r}\| \leq \mathsf{d}_{\mathsf{G}}(\mathsf{q}, \mathsf{r}) \leq t \, \|\mathsf{q} - \mathsf{r}\|,$$

where $\mathsf{d}_{\mathsf{G}}(\mathsf{q}, \mathsf{r})$ is the length of the shortest path in G between q and r (naturally, d_{G} is a metric). The ratio $\mathsf{d}_{\mathsf{G}}(\mathsf{q}, \mathsf{r}) / \|\mathsf{q} - \mathsf{r}\|$ is the ***stretch*** of q and r in G. The ***stretch*** of G is the maximum stretch of any pair of points of P.

Theorem 3.2.1 *Given a n-point set $P \subseteq \mathbb{R}^d$, and parameter $1 \geq \varepsilon > 0$, one can compute a $(1 + \varepsilon)$-spanner of P with $O(n\varepsilon^{-d})$ edges, in $O\left(n \log n + n\varepsilon^{-d}\right)$ time.*

Proof: Let $c \geq 16$ be an arbitrary constant, and set $\delta = \varepsilon / c$. Compute a δ^{-1}-WSPD decomposition using the algorithm of Theorem 3.1.7. 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 $\left\|\mathrm{rep}_u - \mathrm{rep}_v\right\|$, and let G be the resulting graph. Observe, that by the triangle inequality, we have that $\mathsf{d}_{\mathsf{G}}(\mathsf{q}, \mathsf{r}) \geq \|\mathsf{q} - \mathsf{r}\|$, for any $\mathsf{q}, \mathsf{r} \in \mathsf{P}$.

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, that for any pair $z, w \in \mathsf{P}$ such that $\|z - w\| < \|x - y\|$, it holds $\mathbf{d}_{\mathsf{G}}(z, w) \leq (1 + \varepsilon) \|z - w\|$.

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

$$\left\|\mathrm{rep}_u - \mathrm{rep}_v\right\| \leq \mathbf{d}(u, v) + \Delta(u) + \Delta(v) \leq (1 + 2\delta) \|x - y\|$$

and

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

by Theorem 3.1.7 and since $\delta \leq 1/16$. As such, we can apply the induction hypothesis to $\mathrm{rep}_u x$ and $\mathrm{rep}_v y$, implying that

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

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

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

The last step follows by an easy calculation. Indeed, since $c\delta = \varepsilon \leq 1$ and $16\delta \leq 1$ and $c \geq 11$, we have that

$$(1 + 2\delta + 2\varepsilon\delta)(1+\delta) \leq (1+4\delta)(1+\delta) = 1 + 5\delta + 4\delta^2 \leq 1 + 9\delta \leq 1 + \varepsilon,$$

as required. ∎

3.2.2 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 3.2.2 *Given a set P of n points in $\mathbb{R}^d$, one can compute a spanning tree $\mathcal{T}$ of P, such that $w(\mathcal{T}) \leq (1+\varepsilon)w(\mathcal{M})$, where $\mathcal{M}$ is the minimum spanning tree of P, and $w(\mathcal{T})$ is the total weight of the edges of $\mathcal{T}$. This takes $O\left(n\log n + n\varepsilon^{-d}\right)$ time.*

In fact, for any $r \geq 0$ and a connected component C of $\mathcal{M}_{\leq r}$, the set C is contained in a connected component of $\mathcal{T}_{\leq(1+\varepsilon)r}$.

Proof: Compute a $(1+\varepsilon)$-spanner G of P. Let $\mathcal{T}$ be the minimum spanning tree of G. Clearly, $\mathcal{T}$ is the required $(1+\varepsilon)$-approximate MST. Indeed, for any $\mathsf{q}, \mathsf{r} \in \mathsf{P}$, let π_{qr} denote the shortest path between q and r in G. Since G is a $(1+\varepsilon)$-spanner, we have that $w\left(\pi_{\mathsf{qr}}\right) \leq (1+\varepsilon)\|\mathsf{q} - \mathsf{r}\|$, where $w\left(\pi_{\mathsf{qr}}\right)$ denote the weight of π_{qr} in G.

We have that $G' = (P, E)$ is a connected subgraph of G, where

$$E = \bigcup_{(\mathsf{q},\mathsf{r})\in\mathcal{M}} \pi_{uv}$$

and $\mathcal{M} = \mathcal{M}(\mathsf{P})$ is the minimum spanning tree of P. Furthermore,

$$w(G') = \sum_{(\mathsf{q},\mathsf{r})\in\mathcal{M}} w\left(\pi_{\mathsf{qr}}\right) \leq \sum_{(\mathsf{q},\mathsf{r})\in\mathcal{M}} (1+\varepsilon)\|\mathsf{q} - \mathsf{r}\| = (1+\varepsilon)w(\mathcal{M}),$$

since G is a $(1+\varepsilon)$-spanner. It thus follows that $w(\mathcal{M}(G)) \leq w(G') \leq (1+\varepsilon)w(\mathcal{M}(P))$, where $\mathcal{M}(G)$ is the minimum spanning tree of G.

The second claim follows by similar argumentation. ∎

3.2.3 Approximating the Diameter

Lemma 3.2.3 *Given a set* P *of* n *points in* $\mathbb{R}^d$*, one can compute, in* $O\left(n \log n + n\varepsilon^{-d}\right)$ *time, a pair* $u, v \in \mathsf{P}$*, such that* $\|u - v\| \geq (1 - \varepsilon) \operatorname{diam}(P)$.

Proof: Compute a $(4/\varepsilon)$-WSPD of P. As before, we assign for each node u of $\mathcal{T}$ an arbitrary representative point that belongs to P_u. Then, for every pair in the WSPD, compute the distance of the representative point of every pair. Return the pair of representatives in the WSPD farthest away from each other.

To see why it works, consider the pair $\mathsf{q}, \mathsf{r} \in \mathsf{P}$ 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., $\mathsf{q} \in \mathsf{P}_u$ and $\mathsf{r} \in \mathsf{P}_v$). We have that

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

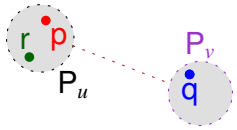
since, by Corollary 3.1.2, $\max(\operatorname{diam}(\mathsf{P}_u), \operatorname{diam}(\mathsf{P}_v)) \leq 2(\varepsilon/4) \|\mathsf{q} - \mathsf{r}\|$. Namely, the distance of the two points output by the algorithm is at least $(1 - \varepsilon) \operatorname{diam}(\mathsf{P})$. ■

3.2.4 Closest Pair

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

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

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



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

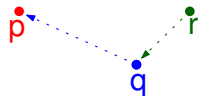
by Theorem 3.1.7 and since $\varepsilon = 1/2$. Thus, $\|\mathsf{p} - \mathsf{r}\| < \|\mathsf{p} - \mathsf{q}\|$, a contradiction to the choice of p and q as the closest pair. Thus, $|\mathsf{P}_u| = |\mathsf{P}_v| = 1$ and $\mathrm{rep}_u = \mathsf{q}$ and $\mathrm{rep}_v = \mathsf{r}$. This implies that the algorithm indeed returns the closest pair.

Theorem 3.2.4 *Given a set* P *of* n *points in* $\mathbb{R}^d$*, 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.

3.2.5 All Nearest Neighbors

Given a set P of n points in $\mathbb{R}^d$, we would like to compute for each point $\mathsf{q} \in \mathsf{P}$, its ***nearest neighbor*** in P (formally, this is the closet point in $\mathsf{P} \setminus \{\mathsf{q}\}$ to q). This is harder than it might seem at first, since this is *not* a symmetrical relationship. Indeed, q might be the nearest neighbor to p, but r might be the nearest neighbor to q.



3.2.5.1 The bounded spread case

Assume P is contained in the unit square, and $\operatorname{diam}(P) \geq 1/4$. Furthermore, let $\Phi = \Phi(P)$ denote the spread of P. Compute a ε^{-1}-WSPD $\mathcal{W}$ of P, for $\varepsilon = 1/4$. Arguing as in the closest pair case, we have that if the nearest neighbor to p is q, then there exists a pair $\{u, v\} \in \mathcal{W}$, such that $\mathsf{P}_u = \{\mathsf{p}\}$ and $\mathsf{q} \in \mathsf{P}_v$. Thus, scan all the pairs $\{u, v\}$ with a singleton as one of their sides (i.e., $|P_u| = 1$), and for each such singleton $\mathsf{P}_u = \{\mathsf{r}\}$, record for r the closest point to it in P_v. Maintain for each point the closet point to it that was encountered.

Analysis. The analysis of this algorithm is slightly tedious, but it reveals some additional interesting properties of WSPD.

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

Lemma 3.2.5 *Let $\mathcal{W}$ be a 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 $\Delta(\overline{\mathrm{p}}(u)) \geq (\varepsilon/2)\mathbf{d}(u, v)$.*

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 3.1.5, 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 $\{u, v'\}$ can not be a generator of $\{u, v\}$. Indeed, if $\{u, v'\}$ was considered by the algorithm than it would have added it to the WSPD, and never created $\{u, v\}$.

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 3.1.5. Using the same argumentation as above, we have that $\{u', v\}$ is a well-separated pair and as such it can not be a generator of $\{u, v\}$.

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

Claim 3.2.6 *For two pairs $\{u, v\}, \{u', v'\} \in \mathcal{W}$ such that $\square_u \subseteq \square_{u'}$, it holds that the interiors of $\square_v$ and $\square_{v'}$ are disjoint.*

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

If u' is ancestor of u, and v is an ancestor of v', then

$$\Delta(u) < \Delta(u') \leq \Delta(\overline{\mathrm{p}}(v')) \leq \Delta(v) \leq \Delta(\overline{\mathrm{p}}(u)) \leq \Delta(u')$$

by Lemma 3.1.5 applied to $\{u', v'\}$ and $\{u, v\}$. Namely, $\Delta(v) = \Delta(u')$. But then, the pair $\{u', v\}$ is a generator of both $\{u, v\}$ and $\{u', v'\}$. But it is impossible that **AlgWSPD**generated both pairs when processing $\{u', v\}$ as can be easily verified.

Similar analysis applies for the case that $u = u'$. ∎

Lemma 3.2.7 *Let* P *be a set n points in* $\mathbb{R}^d$*,* $\mathcal{W}$ *a* ε^{-1}*-WSPD of* P*,* $\ell > 0$ *be a distance, and W be the set of pairs* $\{u, v\} \in \mathcal{W}$ *such that* $\ell \leq \mathbf{d}(u, v) \leq 2\ell$*. Then, point any point* $\mathsf{p} \in \mathsf{P}$*, the number of pairs in W containing* p *is* $O(1/\varepsilon^d)$*.*

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 W contains at most $O(1/\varepsilon^d)$ pairs with nodes that appears along π. Let

$$T = \left\{ v \;\middle|\; u \in \pi, \{u, v\} \in W \right\}.$$

The cells of T are interior disjoint by Claim 3.2.6, 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) $\varepsilon\ell/(4\sqrt{d})$. Clearly, there are $O(1/\varepsilon^d)$ cells of G_r in distance at most 2ℓ from $\square_u$. We account for the nodes $v \in T$, as follows:

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

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

  (a) 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 most $O(1/\varepsilon^d)$ such charges.

  (b) 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 $\mathsf{G}_{r/2}$ in distance at most 2ℓ from $\square_u$, and there are $O(1/\varepsilon^d)$ such cells.

(iii) The case $\Delta(\overline{\mathrm{p}}(v)) < r\sqrt{d}$ is impossible. Indeed, by Lemma 3.1.5, we have $\Delta(\overline{\mathrm{p}}(v)) < r\sqrt{d} \leq \varepsilon\ell/4 = \frac{\varepsilon}{4}\mathbf{d}(u, v)$, a contradiction to Lemma 3.2.5.

We conclude that there are at most $O(1/\varepsilon^d)$ pairs that include p in W. ∎

Lemma 3.2.8 *Let* P *be a set n points in the plane, then one can solve the all nearest neighbor problem, in time* $O(n(\log n + \log \Phi(P)))$ *time, where* Φ *is the spread of* P*.*

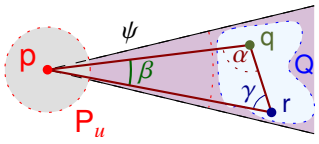
Proof: The algorithm is described above. We only remain with the task of analyzing the running time. For a number $i \in \{0, -1, \ldots, -\lfloor \lg \Phi \rfloor - 4\}$, consider the set of pairs W_i, such that $\{u, v\} \in W_i$, if and only if $\{u, v\} \in \mathcal{W}$, and $2^{i-1} \leq \mathbf{d}(u, v) \leq 2^i$. A point $\mathsf{p} \in \mathsf{P}$ can be scanned at most $O(1/\varepsilon^d) = O(1)$ times because of pairs in W_i by Lemma 3.2.7. As such, a point get scanned at most $O(\log \Phi)$ times overall, which implies the running time bound. ∎

3.2.5.2 All nearest neighbor - the unbounded spread case

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

Lemma 3.2.9 *Let u be a node in the compressed $\mathcal{QT}$ of P, and partition the space around rep_u into cones of angle $\leq \pi/12$. Let ψ be such a cone, and let Q be the set of all points in P which are in distance $\geq 4\,\mathrm{diam}(\mathsf{P}_u)$ from rep_u, and they all lie inside ψ. Let q be the closest point in Q to rep_u. Then, q is the only point in Q that its nearest neighbor might be in P_u.*

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



Since $\|\mathsf{r}-\mathsf{p}\| \geq \|\mathsf{q}-\mathsf{p}\|$, it follows that $\alpha = \angle\mathsf{rqp} \geq \angle\mathsf{qrp} = \gamma$. Now, $\alpha + \gamma = \pi - \beta$, where $\beta = \angle\mathsf{rpq}$. But $\beta \leq \pi/3$, and as such $\alpha \geq (\pi - \angle\mathsf{rpq})/2 \geq \pi/3 \geq \beta$. Namely, α is the largest angle in the triangle $\triangle\mathsf{pqr}$, which implies $\|\mathsf{r}-\mathsf{p}\| \geq \|\mathsf{r}-\mathsf{q}\|$. Namely, q is closer to r than p, and as such p can not serve as the nearest neighbor to r in P.

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

Lemma 3.2.9 implies that we can do a top-down traversal of $\mathcal{QT}(P)$, after computing a 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 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 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 to rep_u. Let R_u be the set of these closest points. Since the number of cones is $O(1)$, it follows that $|\mathsf{R}_u| = O(1)$.

Now, if P_u contains only a single point p, then we compute for any point $\mathsf{q} \in \mathsf{R}_u$ 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.

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. The correctness follows since p is the nearest neighbor to q, then there must be a WSPD pair $\{u, v\}$ such that $\mathsf{P}_v = \{\mathsf{q}\}$ and $\mathsf{p} \in \mathsf{P}_u$. 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 $\mathsf{p} \in \mathsf{P}_z$. In particular, if y is the leaf of the quadtree storing p, then $\mathsf{q} \in \mathsf{R}_y$, which implies that the algorithm computes correctly the nearest neighbor to q.

Theorem 3.2.10 *Given a set P of n points in $\mathbb{R}^d$, one can solve the all nearest neighbor problem in $O(n \log n)$ time.*

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

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

3.3 Bibliographical Notes

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

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

Callahan and Kosaraju [CK95] were inspired by the work of Vaidya [Vai86] on all nearest neighbor problem (i.e., compute for each points in P, their nearest neighbor in P). He defined the fair split tree, and show 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 neighbor algorithm was Clarkson [Cla83] (this was part of his PhD thesis).

Diameter. The algorithm for computing the diameter of Section 3.2.3 can be improved by not constructing pairs that can not improve the (current) diameter, and constructing the underlying tree on the fly together with the diameter. This yields a simple algorithm that works quite will in practice, see [Har01].

All nearest neighbors. Section 3.2.5 is a simplification of the solution for the all k-nearest neighbor 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 neighbor algorithm for bounded spread (Section 3.2.5.1) is from [HM06]. Note, that unlike the unbounded case, this algorithm only use packing arguments for its correctness. Surprisingly, the usage of the Euclidean nature of the underlying space (as done in Section 3.2.5.2) 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.

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 linear number of pairs, and in fact, a node in the quadtree might participate in linear number of pairs. This is described in detail in Callahan 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 in distance 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 kind of tempting to try and

define dimension of a point set by the size of the WSPD it posses. This seems like an interesting direction for future research, as currently little is known about it (to the best of my knowledge).

3.4 Exercises

Exercise 3.4.1 (WSPD Structure.) [5 Points]

(A) Let $\varepsilon > 0$ be 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]

(B) Show, that if we list explicitly the sets forming the WSPD (even if we show each set exactly once) then the total size of such a description is quadratic. (Namely, the implicit representation we use is crucial to achieve efficient representation.)

Exercise 3.4.2 (Number of resolutions that matter.) [4 Points]

Let P be a n-point set in $\mathbb{R}^d$, and consider the set $U = \left\{ i \;\middle|\; 2^i \le \|\mathsf{p}-\mathsf{q}\| \le 2^{i+1},\ \text{for } \mathsf{p},\mathsf{q} \in \mathsf{P} \right\}$. Prove that $|U| = O(n)$ (the constant depends on d). Namely, there are only n different resolutions that "matter".

Exercise 3.4.3 (WSPD and sum of distances.) [5 Points]

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

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

3.5 From previous lectures

Definition 3.5.1 (Canonical square and grid.) 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***.

Bibliography

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

[4] Note, that there is always a WSPD construction such that each node participates in a "small" number of pairs.

[5] Also known as the sum of pairwise distances in the literature, for reasons that I can not fathom.

[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.*, 42:67–90, 1995.

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

[FH05] J. Fischer and S. Har-Peled. Dynamic well-separated pair decomposition made easy. In *CCCG*, pages 235–238, 2005.

[Har01] S. Har-Peled. A practical approach for computing the diameter of a point-set. In *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, pages 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.

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