Visualizing WSPDs and their applications*

Anirban Ghosh ⊠©

School of Computing, University of North Florida, USA

FNU Shariful ⊠©

School of Computing, University of North Florida, USA

David Wisnosky ⊠©

School of Computing, University of North Florida, USA

¹ — Abstract

² Introduced by Callahan and Kosaraju back in 1995, the concept of well-separated pair decomposition

 (WSPD) has occupied a special significance in computational geometry when it comes to solving

 $_{4}$ distance problems in *d*-space. We present an in-browser tool that can be used to visualize WSPDs

 $_{5}$ and several of their applications in 2-space. Apart from research, it can also be used by instructors

for introducing WSPDs in a classroom setting. The tool will be permanently maintained by the
 third author at https://wisno33.github.io/VisualizingWSPDsAndTheirApplications/.

2012 ACM Subject Classification Theory of computation \rightarrow Randomness, geometry and discrete structures

Keywords and phrases well-separated pair decomposition, nearest neighbor, geometric spanners, minimum spanning tree

 ${\small Category} \ {\rm Media} \ {\rm Exposition} \\$

Introduction

Let P and Q be two finite pointsets in d-space and s be a positive real number. We say that P and Q are well-separated with respect to s, if there exist two congruent disjoint balls B_P and B_Q , such that B_P contains the bounding-box of P, B_Q contains the bounding-box 11 of Q, and the distance between B_P and B_Q is at least s times the common radius of B_P 12 and B_Q . The quantity s is referred to as the separation ratio of the decomposition. Using 13 this idea of well-separability, one can define a well-separated decomposition of a pointset 14 (WSPD) [4] in the following way. Let P be a set of n points in d-space and s be a positive 15 real number. A well-separated pair decomposition for P with respect to s is a collection 16 of pairs of non-empty subsets of P, $\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}$ for some integer m 17 (referred to as the size of the WSPD) such that 18

for each i with $1 \le i \le m$, A_i and B_i are well-separated with respect to s, and

for any two distinct points $p, q \in P$, there is exactly one index i with $1 \le i \le m$, such that $p \in A_i, q \in B_i$, or $p \in B_i, q \in A_i$.

Note that in some cases, $m = C(n, 2) = \Theta(n^2)$. Refer to [5, 6, 7] for a detailed discussion on WSPDs and their uses. In this work, we consider WSPDs in 2-space only. Our implementations are based on the algorithms presented in the book by Narasimhan and Smid [6, Chapters 9 and 10]. These algorithms were originally presented in [2, 3, 4] by Callahan and Kosaraju.

© A. Ghosh, F. Shariful, and D. Wisnosky; licensed under Creative Commons License CC-BY 4.0 38th International Symposium on Computational Geometry (SoCG 2022). Editors: Xavier Goaoc and Michael Kerber; Article No. XX; pp. XX:1-XX:4 Leibniz International Proceedings in Informatics LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



^{*} Research on this paper is supported by the NSF award CCF-1947887.

XX:2 Visualizing WSPDs and their applications

²⁶ **2** Algorithms implemented

²⁷ We have implemented the algorithms using the JSXGraph library. Some code segments have
²⁸ been borrowed from the tool presented in [1].

²⁹ 2.1 Constructing WSPDs

Given a pointset P and a positive real number s, a WSPD of P can be constructed using a split-tree. Our implementation is based on the naive quadratic time approach presented in [6]. It accepts P and s, and returns the WSPD pairs in the WSPD decomposition. Refer to Algorithm 1. An advanced linearithmic construction is also presented in [6].

Notations. Let x be a split-tree node. Let S_x denotes the points stored in the subtree rooted at x and R(x) denotes the bounding-box of S_x . Further, $L_{max}(R(x))$ denotes the length of the longer side of R(x).

Algorithm 1 : CONSTRUCTWSPD(P, s > 0)

38 1. Construct a split-tree T on P in the following way:

If |P| = 1, then the split-tree consists of one single node that stores that point. Otherwise, split the bounding-box of P into two rectangles by cutting the longer side of the boundingbox into two equal parts. Let P_1 and P_2 be the two subsets of P that are contained in these two new rectangles. The split-tree for P consists of a root having two subtrees, which are recursively defined for P_1 and P_2 .

- 2. For each internal node u of T, find WSPD pairs using v and w, the left and right child of
 u, respectively, in the following way:
- 46 **a.** Compute S_v , S_w , $L_{max}(R(v))$ and $L_{max}(R(w))$.
- **b.** If S_v, S_w are well-separated with respect to s, then node pair $\{v, w\}$ is a WSPD pair. Otherwise, if $L_{max}(R(v)) \leq L_{max}(R(w))$, recursively find WSPD pairs using v,
- ⁴⁹ LEFTCHILD(w) and then recursively find WSPD pairs using v, RIGHTCHILD(w).
- ⁵⁰ Else, recursively find WSPD pairs using LEFTCHILD(v), w, and then recursively find ⁵¹ WSPD pairs using RIGHTCHILD(v), w.

52 2.2 Applications of WSPDs

- ⁵³ CONSTRUCTION OF t-SPANNERS. Given a pointset P and $t \ge 1$, a t-spanner on P is a ⁵⁴ Euclidean geometric graph G on P such that for every pair of points $p, q \in P$, the length ⁵⁵ of the shortest-path between p, q in G is at most t times the Euclidean distance between ⁵⁶ them. Refer to Algorithm 2. It returns the set of spanner edges and can be implemented
- 57 to run in $O(n \log n)$ time [6].

```
58 Algorithm 2 : Construct-t-Spanner(P, t > 1)
```

```
<sup>59</sup> Let s = 4(t+1)/(t-1). Construct a WSPD of P with separation ratio s. For every pair
<sup>60</sup> (A_i, B_i) of the decomposition do the following: include the edge \{a_i, b_i\} in the spanner where
```

 a_i is an arbitrary point in A_i and b_i is an arbitrary point in B_i .

⁶² FINDING CLOSEST PAIRS. The problem asks to find two distinct points of P whose ⁶³ distance is minimum among the C(n, 2) point pairs. The idea of well-separatedness can

A. Ghosh, F. Shariful, and D. Wisnosky

be used to design an algorithm for this problem. See Algorithm 3. It can be implemented to run in $O(n \log n)$ time [6].

Algorithm 3 : CLOSESTPAIR(P)

⁶⁷ Construct a 2-spanner using Algorithm 2. Since the closest pair is connected by an edge of
 the spanner, find the pair by iterating over all the edges.

FINDING k-CLOSEST PAIRS. It is a generalization of the closest-pair problem. Given a positive integer k such that $k \leq C(n, 2)$, the goal is to find the k closest pairs among the C(n, 2) pairs. See Algorithm 4. It can be implemented to run in $O(n \log n + k)$ time [6].

72 **Algorithm 4** : k-CLOSESTPAIRS(P)

1. Create a WSPD with some s > 0. For every pair (A_i, B_i) in the decomposition, let $R(A_i)$ and $R(B_i)$ be the bounding boxes of A_i and B_i , respectively. Further, by $|R(A_i)R(B_i)|$, we denote the minimum distance between the two bounding-boxes $R(A_i), R(B_i)$. Renumber the *m* pairs in the decomposition such that $|R(A_1)R(B_1)| \leq |R(A_2)R(B_2)| \leq \ldots \leq$ $|R(A_m)R(B_m)|$.

- 78 2. Compute the smallest integer $\ell \geq 1$, such that $\sum_{i=1}^{\ell} |A_i| \cdot |B_i| \geq k$.
- 79 **3.** Let $r := |R(A_\ell)R(B_\ell)|$.

4. Compute the integer ℓ' , which is defined as the number of indices with $1 \le i \le m$, such that $|R(A_i)R(B_i)| \le (1 + 4/s)r$.

⁸² 5. Compute the set *L* consisting of all pairs $\{p, q\}$ for which there is an index *i* with $1 \le i \le \ell'$, ⁸³ such that $p \in A_i, q \in B_i$ or $q \in A_i, p \in B_i$.

- ⁸⁴ **6.** Compute and return the k smallest distances determined by the pairs in the set L.
- FINDING ALL-NEAREST NEIGHBORS. In this problem, for every point p in P, we need to find its nearest neighbor q in $P \setminus \{p\}$. Refer to Algorithm 5 for a description of the algorithm. It can be implemented to run in $O(n \log n)$ time [6].
- **Algorithm 5** : AllNearestNeighbors(P)

⁸⁹ Choose s > 2 and obtain the pairs of WSPD. For every $p \in P$, compute its nearest neighbor ⁹⁰ in the following way: Find all such pairs of the WSPD, for which at least one of their sets is ⁹¹ a singleton containing p. For every such pair (A_i, B_i) , if $A_i = \{p\}$, then $S_p = S_p \cup B_i$, else if ⁹² $B_i = \{p\}$, then $S_p = S_p \cup A_i$. The nearest neighbor of p is the point in S_p closest to p (found ⁹³ by exhaustive search).

```
<sup>94</sup> = t-APPROXIMATE MINIMUM SPANNING TREES. Let t > 1, be a real number. A tree
<sup>95</sup> connecting the points of P is called a t-approximate minimum spanning tree of P, if its
<sup>96</sup> weight is at most t times the weight of the Euclidean minimum spanning tree of P. Refer
<sup>97</sup> to Algorithm 6. In d-space, it runs in O(n \log n + n/(t-1)^d) time [6].
```

```
98 Algorithm 6 : t-ApproximateMinimumSpanningTree(P, t > 1)
```

⁹⁹ Compute the *t*-spanner G using Algorithm 2. Using Prim's algorithm compute a minimum ¹⁰⁰ spanning tree T of G. Return T.

XX:4 Visualizing WSPDs and their applications

101		References
102	1	Fred Anderson, Anirban Ghosh, Matthew Graham, Lucas Mougeot, and David Wisnosky.
103		An interactive tool for experimenting with bounded-degree plane geometric spanners (media
104		exposition). In 37th International Symposium on Computational Geometry (SoCG 2021).
105		Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.
106	2	Paul B Callahan and S Rao Kosaraju. Faster algorithms for some geometric graph problems
107		in higher dimensions. In SODA, volume 93, pages 291–300, 1993.
108	3	Paul B Callahan and S Rao Kosaraju. Algorithms for dynamic closest pair and n -body
109		potential fields. In SODA, volume 95, pages 263–272, 1995.
110	4	Paul B Callahan and S Rao Kosaraju. A decomposition of multidimensional point sets with
111		applications to k-nearest-neighbors and n-body potential fields. Journal of the ACM (JACM),
112		42(1):67-90, 1995.
113	5	Sariel Har-Peled. Geometric approximation algorithms. Number 173. American Mathematical
114		Soc., 2011.
115	6	Giri Narasimhan and Michiel Smid. Geometric spanner networks. Cambridge University Press,
116		2007.
117	7	Michiel Smid. The well-separated pair decomposition and its applications. In Handbook of

Approximation Algorithms and Metaheuristics, pages 71–84. Chapman and Hall/CRC, 2018.