On the Power of the Semi-Separated Pair Decomposition

Mohammad Ali Abam[1]

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.

Paz Carmi[2]

School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada.

Mohammad Farshi[2,*]

Department of Computer Science, Yazd University, Yazd, Iran.

Michiel Smid[2]

School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada.

Abstract

A Semi-Separated Pair Decomposition (SSPD), with parameter $s > 1$, of a set $S \subset \mathbb{R}^d$ is a set $\{(A_i, B_i)\}$ of pairs of subsets of S such that for each i, there are balls D_{A_i} and D_{B_i} containing A_i and B_i respectively such that $d(D_{A_i}, D_{B_i}) \geq s \cdot \min(\text{radius}(D_{A_i}), \text{radius}(D_{B_i}))$, and for any two points $p, q \in S$ there is a unique index i such that $p \in A_i$ and $q \in B_i$ or vice-versa. In this paper, we use the SSPD to obtain the following results: First, we consider the construction of geometric t-spanners in the context of imprecise points and we prove that any set $S \subset \mathbb{R}^d$ of n imprecise points, modeled as pairwise disjoint balls, admits a t-spanner with $\mathcal{O}(n \log n/(t-1)^d)$ edges that can be computed in $\mathcal{O}(n \log n/(t-1)^d)$ time. If all balls have the same radius, the number of edges reduces to $\mathcal{O}(n/(t-1)^d)$. Secondly, for a set of n points in the plane, we design a query data structure for half-plane closest-pair queries that can be built in $\mathcal{O}(n^2 \log^2 n)$ time using $\mathcal{O}(n \log n)$ space and answers a query in $\mathcal{O}(n^{1/2+\varepsilon})$ time, for any $\varepsilon > 0$. By reducing the preprocessing time to $\mathcal{O}(n^{1+\varepsilon})$ and using $\mathcal{O}(n \log^2 n)$ space, the query can be answered in $\mathcal{O}(n^{3/4+\varepsilon})$ time. Moreover, we

[*]Corresponding author.
Email addresses: `abam@sharif.ir` (Mohammad Ali Abam), `paz@cg.scs.carleton.ca` (Paz Carmi), `mfarshi@yazd.ac.ir` (Mohammad Farshi), `michiel@scs.carleton.ca` (Michiel Smid)
[1]MAA was supported by School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran.
[2]PC, MF and MS were supported by Natural Sciences and Engineering Research Council (NSERC) of Canada.

improve the preprocessing time of an existing axis-parallel rectangle closest-pair query data structure from quadratic to near-linear. Finally, we revisit some previously studied problems, namely spanners for complete k-partite graphs and low-diameter spanners, and show how to use the SSPD to obtain simple algorithms for these problems.

Keywords: semi-separated pair decomposition, closest-pair query, imprecise spanners, spanners for complete k-partite graphs

1. Introduction

Background. The Well-Separated Pair Decomposition (WSPD) introduced by Callahan and Kosaraju (Callahan and Kosaraju, 1995) has found numerous applications in proximity problems (Narasimhan and Smid, 2007, Chapter 10). A WSPD for a point set $S \subset \mathbb{R}^d$ with respect to a constant $s > 1$ is a set of pairs $\{(A_i, B_i)\}_i$ where (i) $A_i, B_i \subset S$, (ii) A_i and B_i are s-well-separated, i.e., there are balls D_{A_i} and D_{B_i} containing A_i and B_i, respectively, such that $d(D_{A_i}, D_{B_i}) \geq s \cdot \max(\text{radius}(D_{A_i}), \text{radius}(D_{B_i}))$, and (iii) for any two points $p, q \in S$ there is a unique index i such that $p \in A_i$ and $q \in B_i$ or vice-versa. Callahan and Kosaraju showed that a WSPD containing $\mathcal{O}(s^d n)$ pairs can be constructed in $\mathcal{O}(s^d n + n \log n)$ time. Although they showed that $\sum \min(|A_i|, |B_i|) = \mathcal{O}(n \log n)$, the summation $\sum(|A_i| + |B_i|)$, the so-called *weight* of the WSPD, can be $\Theta(n^2)$. This disadvantage led Varadarajan (Varadarajan, 1998) to define the Semi-Separated Pair Decomposition (SSPD).

An SSPD is defined as a WSPD, except for the condition (ii) which is relaxed to the requirement that A_i and B_i are s-semi-separated, i.e., there are balls D_{A_i} and D_{B_i} containing A_i and B_i, respectively, such that $d(D_{A_i}, D_{B_i}) \geq s \cdot \min(\text{radius}(D_{A_i}), \text{radius}(D_{B_i}))$. Varadarajan (Varadarajan, 1998) showed how to compute an SSPD of weight $\mathcal{O}(n \log^4 n)$ for a set of n points in the plane in $\mathcal{O}(n \log^5 n)$ time and used the decomposition to solve the min-cost perfect-matching problem. Recently, Abam et al. (Abam et al., 2009) presented an algorithm that improves the construction time to $\mathcal{O}(n \log n)$ and the weight to $\mathcal{O}(n \log n)$; in Abam et al. (2011), the same bounds were obtained in $\mathbb{R}^d$. It follows from results by Hansel (Hansel, 1964) that any SSPD of any set of n points has weight $\Omega(n \log n)$ —see Bollobás and Scott (Bollobás and Scott, 2007) as well.

Abam et al. (Abam et al., 2009) used the SSPD to compute a region fault-tolerant t-spanner in the plane, which is a geometric t-spanner, as defined next, and remains a t-spanner after everything inside a half-plane fault region is removed from the spanner.

Let $G = (S, E)$ be a geometric graph on a set S of n points in $\mathbb{R}^d$. That is, G is an edge-weighted graph where the weight of an edge $(p, q) \in E$ is equal to $|pq|$, the Euclidean distance between p and q. The distance in G between two points p and q, denoted by $d_G(p, q)$, is defined as the length of a shortest (that is, minimum-weight) path between p and q in G. The graph G

is called a (geometric) *t-spanner,* for some $t \geq 1$, if for any two points $p, q \in S$ we have $d_G(p, q) \leq t \cdot |pq|$. We define a *$t$-path* between p and q to be any path between p and q having length at most $t \cdot |pq|$. Geometric spanners have received a lot of attention in the past few years—see the book by Narasimhan and Smid (Narasimhan and Smid, 2007) and survey papers Eppstein (2000); Gudmundsson and Knauer (2007); Smid (2000) for more details. Obviously the complete graph is a t-spanner for any point set and any $t \geq 1$, but a desirable one would be a spanner providing short paths between its nodes, while not containing too many edges. Other desirable properties of a t-spanner is low total length of edges and low maximum degree. Several algorithms are known that compute, in near-linear time, a t-spanner of any given point set that has one or more desirable properties—see Narasimhan and Smid (2007).

Our results. In this paper, we present more applications of the SSPD and show how powerful the SSPD can be:
(i) We consider geometric t-spanners in the context of imprecise points. We model each imprecise point as a ball that specifies the possible location of the point. For a set of n pairwise disjoint imprecise points in $\mathbb{R}^d$, for a constant d, we compute a geometric t-spanner with $\mathcal{O}(n \log n/(t-1)^d)$ edges such that regardless of the position of each point in its associated ball, it remains a t-spanner. Moreover, we improve the number of edges to $\mathcal{O}(n/(t-1)^d)$ if the associated balls have the same radius.
(ii) We present a query data structure for the half-plane closest-pair query problem that uses $\mathcal{O}(n \log^2 n)$ space and can be computed in $\mathcal{O}(n^{1+\varepsilon})$ time and answers a query in $\mathcal{O}(n^{3/4+\varepsilon})$ time, where $\varepsilon > 0$ is an arbitrary constant. By increasing the pre-processing time to $\mathcal{O}(n^2 \log^2 n)$ and using $\mathcal{O}(n \log n)$ space, we achieve $\mathcal{O}(n^{1/2+\varepsilon})$ query time. We also improve the pre-processing time of the axis-parallel rectangle closest-pair query data structure of Gupta et al. (2008) from quadratic to near-linear without affecting the query time and space bound.
(iii) We revisit some previously studied problems, specifically spanners for k-partite graphs (Bose et al., 2008) and low-diameter spanners (Arya et al., 1995, 1994), and show how to use the SSPD to obtain simple algorithms for these problems. Here, we just emphasize on the simplicity of the algorithms; we do not improve the existing results.

2. Spanners for Imprecise Points

Computational geometers traditionally assume that input data, such as points, are precise. However, in the real-world, the input comes from measuring devices that are subject to finite precision. Therefore, the input data given to an algorithm is imprecise and running the algorithm on the input may lead to incorrect output. One solution is to design algorithms that explicitly compute with imprecise data that can be modeled in different ways. One possible model, for data that consists of points, is to consider each point as a region. This region represents all possible locations where the point might be. Given a collection of such imprecise points, one can then ask questions about these points. What is their convex hull? What is their Voronoi diagram/Delaunay triangulation? These questions were recently studied— see Löffler and Snoeyink (2008); van Kreveld and Löffler (2008)— and here we consider one more interesting question.

Let $\mathfrak{D} = \{D_1, \ldots, D_n\}$ be a set of n regions in $\mathbb{R}^d$ and let $t > 1$ be a real number. Is it possible to construct a graph $G = (\mathfrak{D}, \mathfrak{E})$, such that for any set $\mathcal{S} = \{p_1, \ldots, p_n\}$ of points, with $p_i \in D_i$ for $1 \leq i \leq n$, the graph $G = (\mathcal{S}, \mathcal{E})$, where $\mathcal{E} = \{(p_i, p_j) | (D_i, D_j) \in \mathfrak{E}\}$ is a t-spanner? We call the graph G a t-spanner for the set $\mathfrak{D}$. In this section we answer this question affirmatively for the case when the regions are balls.

In fact, we use the common model of expressing imprecise points where each imprecise point p_i is modeled by a ball $D_i = (c_i, r_i)$, where c_i and r_i are the center and the radius of D_i, in $\mathbb{R}^d$. We assume balls are pairwise disjoint. Otherwise, for any two overlapping balls there must be an edge in a t-spanner, as we know the edge between the closest pair is always in the t-spanner for $t < 2$, and therefore the size of the t-spanner depends on the number of overlapping balls. This assumption is not unrealistic as constructing road network is a common applicaton of t-spanners where cities are regions and they do not overlap.

2.1. Balls with Similar Sizes

We first consider the case when all balls are unit-balls. We can easily extend the results to the case when all balls have similar sizes. Our spanner construction is based on the WSPD approach (Callahan and Kosaraju, 1993) that works as follows. It computes a WSPD of the point set with respect to a constant s, and then for each pair (A, B) in the WSPD, it adds an edge between an arbitrary point from A and an arbitrary point from B. Choosing an appropriate value s based on t leads us to a t-spanner.

The above construction is applicable to an imprecise point set if we are able to construct a WSPD of the imprecise point set, i.e., regardless of the positions of the points in their associated balls, the pairs in the decomposition remain s-well-separated. The following lemma states that it is possible to obtain a WSPD of imprecise points using a WSPD of the center points.

Lemma 1. *Let $\{(A_i, B_i) | 1 \leq i \leq m\}$ be a WSPD for the set $\{c_1, \ldots, c_n\}$ of center points, with respect to $s' = 2s + 2$. Let $\mathcal{S} = \{p_1, \ldots, p_n\}$ be a set of points, where $p_j \in D_j$, for $1 \leq j \leq n$. For $1 \leq i \leq m$, let $A'_i = \{p_j | c_j \in A_i\}$ and $B'_i = \{p_j | c_j \in B_i\}$. Then $\{(A'_i, B'_i) | 1 \leq i \leq m\}$ is a WSPD for $\mathcal{S}$ with respect to s.*

Proof. Let (A, B) be an s'-well-separated pair. There are two balls D_A and D_B containing the points in A and B, respectively, and $d(D_A, D_B) \geq (2s+2) \cdot \max(\text{radius}(D_A), \text{radius}(D_B))$. If $\max(\text{radius}(D_A), \text{radius}(D_A)) < 1$, then the disjointness of the balls D_i implies that A and B are singletons, which implies that (A', B') is an s-well-separated pair. Otherwise, let D'_A (D'_B) be a ball with radius $\text{radius}(D_A) + 1$ ($\text{radius}(D_B) + 1$) co-centered with D_A (D_B). Since $|p_i c_i| \leq 1$, from $c_i \in D_A$ we can conclude that $p_i \in D'_A$. The same property holds for B. Therefore D'_A and D'_B contain all points in A' and B', respectively. Now it suffices to show that $d(D'_A, D'_B) \geq s \cdot \max(\text{radius}(D'_A), \text{radius}(D'_B))$. We have

$$\begin{aligned} d(D'_A, D'_B) &= d(D_A, D_B) - 2 \\ &\geq s' \cdot \max(\text{radius}(D_A), \text{radius}(D_B)) - 2 \\ &\geq (s' - 2) \cdot \max(\text{radius}(D_A), \text{radius}(D_B)) \\ &\geq 2s \cdot \max(\text{radius}(D_A), \text{radius}(D_B)) \\ &\geq s \cdot (\max(\text{radius}(D_A), \text{radius}(D_B)) + 1) \\ &\geq s \cdot \max(\text{radius}(D'_A), \text{radius}(D'_B)). \end{aligned}$$

■

Theorem 2. *For any set $\mathfrak{D} = \{D_1, \ldots, D_n\}$ of n imprecise points in $\mathbb{R}^d$ modeled as pairwise disjoint balls with similar sizes and any $t > 1$, there is a t-spanner with $\mathcal{O}(n/(t-1)^d)$ edges that can be computed in $\mathcal{O}(n/(t-1)^d + n \log n)$ time.*

Proof. Let $s = \frac{4(t+1)}{t-1}$ and $s' = 2s + 2$. Let $\{(A_i, B_i) | 1 \leq i \leq m\}$ be a WSPD with respect to s' for center points of size $m = \mathcal{O}(s'^d n)$. Initialize $\mathfrak{E} = \emptyset$. For each $1 \leq i \leq m$ add edge (D_j, D_k) to $\mathcal{E}$, where D_j is a ball with center $c_j \in A_i$ and D_k is a ball with center $c_k \in B_i$. Let $G = (\mathfrak{D}, \mathfrak{E})$ be the resulting graph.

Let $\mathcal{S} = \{p_1, \ldots, p_n\}$ be a set where $p_i \in D_i$, for $1 \leq i \leq n$, and let $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{E} = \{(p_i, p_j) | (D_i, D_j) \in \mathfrak{E}\}$. It follows from Lemma 1 and Callahan and Kosaraju (1993) that $\mathcal{G}$ is a t-spanner for $\mathcal{S}$ and the time complexity of the algorithm is $\mathcal{O}(n/(t-1)^d + n \log n)$. ■

2.2. Balls with Arbitrary Sizes

When the sizes of the balls vary greatly, we cannot simply construct a WSPD of the points p_i using a WSPD of the center points c_i. Hence, a more sophisticated approach is needed. As we will see, the SSPD comes handy here. The overall idea is to construct an SSPD of the points $\{p_i\}_i$ using an SSPD of the points $\{c_i\}_i$ and then construct a t-spanner using the SSPD of the points $\{p_i\}_i$.

Lemma 3. *Let $\{(A_i, B_i) | 1 \leq i \leq m\}$ be a SSPD for the set $\{c_1, \ldots, c_n\}$ of center points, with respect to $s' = 3s + 3$. Let $\mathcal{S} = \{p_1, \ldots, p_n\}$ be a set of points, where $p_j \in D_j$, for $1 \leq j \leq n$. For $1 \leq i \leq m$, let $A'_i = \{p_j | c_j \in A_i\}$ and $B'_i = \{p_j | c_j \in B_i\}$. Then $\{(A'_i, B'_i) | 1 \leq i \leq m\}$ is a SSPD for $\mathcal{S}$ with respect to s.*

Proof. Let (A, B) be an s'-semi-separated pair in the SSPD of the points $\{c_i\}_i$. Let A' (resp. B') be the set containing the points $p_i \in D_i$ corresponding to the points $c_i \in A$ (resp. $c_i \in B$). Since (A, B) is an s'-semi-separated pair, there are two balls D_A and D_B containing all points in A and B, respectively, such that $d(D_A, D_B) \geq s' \cdot \min(r_A, r_B))$, where $r_A = \text{radius}(D_A)$ and $r_B = \text{radius}(D_B)$. Without loss of generality, assume that $r_A \leq r_B$. If $\text{radius}(D_i) \geq 2 \cdot r_A$, for

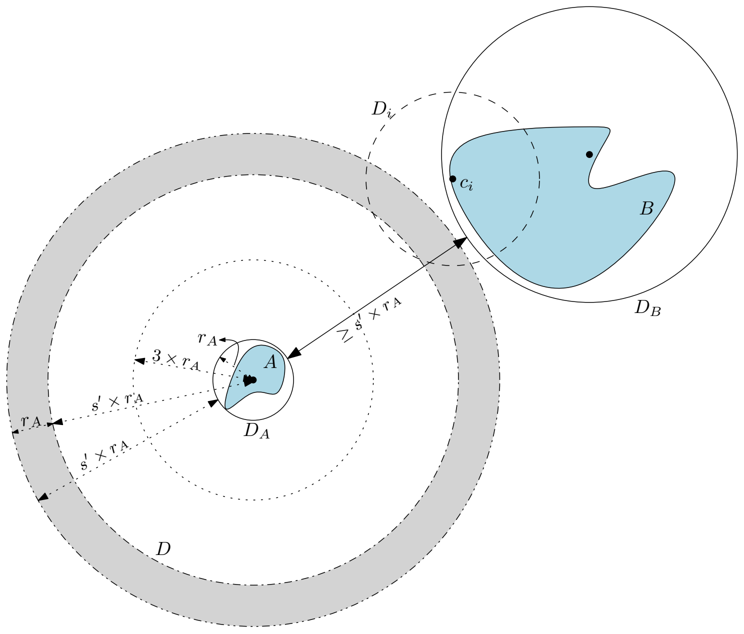

Figure 1: Illustration of Lemma 3.

some i such that $c_i \in A$, the disjointness of the balls implies that A, and as a consequence A', is a singleton and therefore (A', B') is an s-semi-separated pair.

Otherwise, assume that for any point $c_i \in A$, radius$(D_i) < 2 \cdot r_A$. Therefore, every point p_i corresponding to the point $c_i \in A$ must lie in the ball co-centered with D_A and having radius $3 \cdot r_A$.

Let D be the ball co-centered with D_A and radius $s' \cdot r_A$, see Figure 1. Using a packing argument, it can be shown that the number of points $c_i \in B$ whose associated balls intersect D and have radius greater than r_A is bounded by $\mathcal{O}(d^{d/2}s^{d-1})$, see (Duncan et al., 2001, Lemma 3.2). For each such point c_i, $(A', \{p_i\})$ is an s-semi-separated pair. For the remaining points c_i, the corresponding point p_i is at least $(s'-1) \cdot r_A$ away from D_A. This implies that these points are at least $(s'-3) \cdot r_A$ away from the points in A'; the latter points are inside a ball with radius $3 \cdot r_A$. This all together implies that these points and A' are s-semi-separated, because $(s'-3)/3 = s$. Note that each pair (A, B) produces a constant number, or more precisely $\mathcal{O}(s^d)$, of pairs each of which has linear size based on A and B and therefore the weight of the generated SSPD is $\mathcal{O}(s^{2d} n \log n)$. ■

Our spanner construction is as follows. First we use Lemma 3 to compute an SSPD $\mathcal{S}$ of the points $\{p_i\}_i$ with respect to $s = 4/(t-1)$. Then, for each pair $(A, B) \in \mathcal{S}$, assuming radius$(D_A) \leq$ radius(D_B), we select an arbitrary point from A and connect it by an edge to every other point in $A \cup B$. The number of edges added to the spanner is at most $\sum_{(A,B)\in\mathcal{S}}(|A|+|B|)$ which is $\mathcal{O}(n\log n)$ based on the property of the SSPD. We claim that this gives a t-spanner. To prove this, let p and q be two arbitrary points. There is a pair $(A,B)\in\mathcal{S}$ such that $p\in A$ and $q\in B$ or vice-versa. Assume that $\text{radius}(D_A)\leq\text{radius}(D_B)$, $p\in A$, and $q\in B$. Based on our construction, both p and q are connected to a point w in A—note that w can be p. Therefore the length of the path between p and q in the graph is at most $|pw|+|wq|$, which can be bounded as follows:

$$|pw|+|wq|\leq 2|pw|+|pq|\leq 4\,\text{radius}(D_A)+|pq|\leq(t-1)|pq|+|pq|\leq t\cdot|pq|.$$

This shows that the path is a t-path between p and q.

Theorem 4. *For any set of n imprecise points in $\mathbb{R}^d$ modeled as pairwise disjoint balls and any $t>1$, there is a t-spanner with $\mathcal{O}(n\log n/(t-1)^{2d})$ edges that can be computed in $\mathcal{O}(n\log n/(t-1)^{2d})$ time.*

3. Range Closest-Pair Query

The range searching problem is a well-studied problem in computational geometry. In such a problem, we are given a set of geometric objects, such as points or line segments, and want to pre-process the set into a data structure such that we can report the objects in a query region quickly—see the survey by Agarwal and Erickson (Agarwal and Erickson, 1999). However, in several applications, we need more information about the objects in the query area, for example the closest pair or the proximity of these objects. For this kind of queries, a so-called *aggregation function* can be defined to satisfy the property we are looking for. This *range-aggregate* query problem has been studied in recent years in both the computational geometry (Nievergelt and Widmayer, 2000) and the database communities (Tao and Papadias, 2004).

The range-aggregate query problem for the case when ranges are axis-parallel rectangles and the aggregation function is the closest pair, was first considered by Shan et al. (Shan et al., 2003). They proposed an algorithm and showed that it works well in practice, but no theoretical bound was provided. Later Gupta (Gupta, 2006) gave a data structure with constant query time using $\mathcal{O}(n)$ space for points in $\mathbb{R}$. For points in the plane, their structure answers a query in $\mathcal{O}(\log^3 n)$ time and uses $\mathcal{O}(n^2\log^3 n)$ space. Later, Sharathkumar and Gupta (Sharathkumar and Gupta, 2007) improved the space in the $2D$ case to $\mathcal{O}(n\log^3 n)$ while guaranteeing the same query time. Recently, Gupta et al. Gupta et al. (2008) improved the query time to $\mathcal{O}(\log^2 n)$ using $\mathcal{O}(n\log^5 n)$ space. It is unknown whether the data structures in Gupta et al. (2008); Sharathkumar and Gupta (2007) can be built in sub-quadratic time.

In this section, we first present a data structure for range closest-pair query problem when ranges are half-planes. Then, we show how to modify Gupta et al.'s data structure in Gupta et al. (2008) such that it can be built in near-linear time without affecting the query time and space bound.

3.1. Half-Plane Closest-Pair Query

Let S be a set of n points in the plane. We first start investigating which pairs of points can be a closest pair for some half-plane. Let G be the graph with vertex set S where p and q are connected if and only if (p, q) is a closest pair in $S \cap h$ for some half-plane h. The following lemma states that the number of such closest pairs is $\mathcal{O}(n)$, even though the number of "different" half-planes is $\Theta(n^2)$.

Lemma 5. *The graph G defined above is plane.*

Proof. For the sake of contradiction, assume that (p, q) and (r, s) properly intersect, where (p, q) and (r, s) are the closest pairs inside the half-planes h_1 and h_2, respectively. It is easy to see that h_1 contains at least one of the points r and s. Assume that r is inside h_1. Since (p, q) is the closest pair inside h_1, $|pr|$ and $|qr|$ are at least $|pq|$. The same argument holds for (r, s). Under the assumption that p is in h_2, we can conclude that $|pr|$ and $|ps|$ are at least $|rs|$. This all together implies that $|pq| + |rs| \leq |ps| + |rq|$. On the other hand, $|pq| + |rs| > |ps| + |rq|$, since (p, q) and (r, s) properly intersect. This contradiction implies that the graph G is plane. ■

We describe our data structure under the assumption that G is available to us. Later, we will explain how to construct G. We construct a half-plane segment-reporting data structure for the edges of G, which is a multi-level partition tree—see (de Berg et al., 2008, Section 16.2). This data structure stores n segments not sharing any endpoint in such a way that the segments inside the half-plane query can be reported as the union of $\mathcal{O}(n^{1/2+\varepsilon})$ disjoint canonical subsets. The data structure uses $\mathcal{O}(n \log n)$ space and can be constructed in $\mathcal{O}(n^{1+\varepsilon})$ time. We also pre-compute the closest pair for each canonical subset of nodes in the second level of the tree to be able to report the closest pair without visiting all the edges in the query region.

The assumption that the segments are not sharing any endpoint can be relaxed to the assumption that each endpoint can be adjacent to at most a constant number of segments. Indeed, by such an assumption, the size of the associated partition tree with a node v in the first level is still proportional to $|S(v)|$, where $S(v)$ is the set of points stored at the subtree rooted at v. This is the key property in the analysis of the space and time complexity. Unfortunately, this assumption does not hold in our graph G, as we can simply find a configuration of n points such that the maximum degree in G is $\Theta(n)$.

A possible way to resolve this problem is to use the *simulation of simplicity* technique of Edelsbrunner and Mücke (1990). Using this technique, we get a set of segments that do not share any endpoint. However, it is not clear how can we construct the multi-level partition tree on the new point set. Therefore, we give a simple and direct way to handle shared endpoints.

We first make the graph G directed such that the out-degree of each vertex is constant. This can be performed as follows. We select a set S_1 of $n/2$ vertices whose degrees are at most 12—this is always possible, since the degree sum in any plane graph is at most 6 times the number of vertices. For each edge (p, q), if both p and q are in S_1 we give this edge an arbitrary direction. If one of them is in S_1, say p, we make the edge (p, q) directed in the direction $\overrightarrow{pq}$. We remove every directed edge as well as the vertices in S_1 from the graph and recurse on the remaining graph. At the end, we have a directed graph $\overrightarrow{G}$ such that the out-degree of each node is at most 12.

Now, given the graph $\overrightarrow{G}$, for each node v in the first level of the multi-level partition tree, we look at the edges going out from $S(v)$. Since the number of such edges is proportional to $|S(v)|$, the same query-time bound can be obtained.

One easy way of computing G is to compute the closest pair for all $2 \cdot \binom{n}{2}$ possible half-planes which takes $\mathcal{O}(n^3 \log n)$ time. Unfortunately it seems difficult to compute G in near-linear time. Hence, we introduce a new graph G' with $\mathcal{O}(n \log n)$ edges that contains G as a subgraph and can be computed in near-linear time. To define G', we use the convex region fault-tolerant t-spanner of the points, as introduced by Abam et al. (Abam et al., 2009). This graph has the property that after removing all vertices and edges that are inside a convex fault region, what remains is a t-spanner of the complete graph minus the vertices and edges in the convex fault region. They used an SSPD to construct a convex region fault-tolerant t-spanner containing $\mathcal{O}(n \log n)$ edges in $\mathcal{O}(n \log^2 n)$ time. When the fault regions are half-planes, what remains from the graph is a t-spanner of the remaining points due to the fact that the line segment between any two points outside the half-plane fault region does not touch the fault region. Since any t-spanner, for $t < 2$, contains the closest pair as an edge, we set G' to be a region fault-tolerant t-spanner for some $t < 2$.

There are two possibilities of using G': (i) use G' instead of G in the above construction and (ii) use G' to compute G faster. Next we look at each of them more precisely.

(i) Using G' instead of G in our structure will obviously affect the asymptotic complexity of the space bound by a factor of $\mathcal{O}(\log n)$. Moreover, since we cannot make G' directed such that the out-degree of each vertex is bounded, we are unable to obtain the same query-time bound. We can show, however, that the query time is $\mathcal{O}(n^{3/4+\varepsilon})$: Searching in the first level of the multi-level partition tree boils down to $\mathcal{O}(n^{1/2+\varepsilon})$ associated partition trees that are disjoint and whose total size is $\mathcal{O}(n \log n)$. If x is the size of one of the associated trees, searching in the associated tree takes $\mathcal{O}(x^{1/2+\varepsilon})$ time. By the Cauchy-Schwarz inequality, we know that $\sum_{i=1}^{m} \sqrt{x_i}/m \leq \sqrt{\sum_{i=1}^{m} x_i/m}$. Therefore, the total search costs $\mathcal{O}(n^{3/4+\varepsilon})$—note that $m = \mathcal{O}(n^{1/2+\varepsilon})$ and $\sum_{i=1}^{m} x_i = \mathcal{O}(n \log n)$.

(ii) We can construct G from G' as follows. We sort all edges of G' by their lengths and process them in ascending order. Initially, we set G to be the graph on the point set whose edge set is empty. Let e be the edge of G' to be processed. We check in linear time whether it intersects any of the current edges of G. If so, we ignore e. Otherwise, we perform two rotational sweeps around the endpoints of e, in $\mathcal{O}(n \log n)$ time, to see whether there is a half-plane containing e that does not contain any edge in the current graph G. If so, e is inserted into G, otherwise, we ignore e. Since we process $\mathcal{O}(n \log n)$ edges, each of which takes $\mathcal{O}(n \log n)$ time, the total construction time is $\mathcal{O}(n^2 \log^2 n)$.

Theorem 6. *Let S be a set of n points in the plane. For any $\varepsilon > 0$, there is a data structure for S*

(i) of size $\mathcal{O}(n \log^2 n)$ that can be constructed in $\mathcal{O}(n^{1+\varepsilon})$ time and answers a half-plane closest-pair query in $\mathcal{O}(n^{3/4+\varepsilon})$ time; or

(ii) of size $\mathcal{O}(n \log n)$ that can be constructed in $\mathcal{O}(n^2 \log^2 n)$ time and answers a half-plane closest-pair query in $\mathcal{O}(n^{1/2+\varepsilon})$ time.

Remark 1. In a recent work by Abam and Har-Peled (Abam and Har-Peled, 2012), new constructions of SSPD are introduced with the following additional property: each point appears in poly-logarithmic number of sets in SSPD. Using the new constructions, one can construct the graph G' such that the out-degree of each node is poly-logarithmic.[3] This improves the half-plane closest-pair query time in Theorem 6(i) to $\mathcal{O}(n^{1/2+\varepsilon})$.

Remark 2. There is a recent work by T. M. Chan (Chan (2012)) that improves the preprocessing time of the partition tree to logarithmic, but it is randomized. This improves the expected preprocessing time of Theorem 6(i) to $\mathcal{O}(n \log n)$.

Remark 3. Like other data structures using partition trees, it is possible to obtain a trade-off between query time and space. Indeed, for any $n \le m \le n^2$, there is a data structure of size $\mathcal{O}(m^{1+\varepsilon})$ and $\mathcal{O}(n^{1+\varepsilon}/m^{1/2})$ query time.

3.2. Axis-Parallel Rectangle Closest-Pair Query

We now consider the axis-parallel rectangle closest-pair query. As mentioned above, Gupta et al. (Gupta et al., 2008) presented a data structure of size $\mathcal{O}(n \log^5 n)$ and query time $\mathcal{O}(\log^2 n)$. It is unknown whether their structure can be built in subquadratic time. Their data structure works as follows: They first construct a data structure to answer closest-pair queries for two-sided queries (vertical/horizontal strips and quadrants). To do that, they pre-compute a graph G with vertex set S whose edges are closest pair for some two-sided region. They show that G has linear size for quadrants and $\mathcal{O}(n \log n)$ size for vertical/horizontal strips; however, it is unknown how to compute G quickly. For three- and four-sided queries, they use the data structure for two-sided queries together with some additional information that can be computed in near-linear time. Therefore, the time-consuming ingredient of their structure is computing the graph G.

[3]The fault-tolerant t-spanner G' constructed based on the algorithm given in (Abam et al., 2009, section 3.1) is the union of $\mathcal{O}(n)$ plane subgraphs where each subgraph is a graph defined over a pair of the SSPD. Moreover, each pair is involved in a constant number of such subgraphs. Since each subgraph is plane, we can make it directed such that the out-degree of each vertex is at most 6. Since each point appears in poly-logarithmic pairs in the new constructions of SSPD, the out-degree of each vertex in G' is then poly-logarithmic.

As in the previous section, we introduce a graph G' that has $\mathcal{O}(n \log n)$ edges, including all edges of G. We use G' instead of G. The graph G' indeed is a kind of t-spanner that we call *local t-spanner.*

A geometric t-spanner G is an F-local spanner, for a region F in the plane, if the part of G that is completely inside F is a t-spanner of the points inside F. For a family $\mathcal{F}$ of regions, we call a graph G an $\mathcal{F}$-local t-spanner, if for any region $F \in \mathcal{F}$ the graph G is an F-local t-spanner. As an example, any convex region fault-tolerant t-spanner is an $\mathcal{H}$-local t-spanner, where $\mathcal{H}$ is the family of half-planes. We will show that there are $\mathcal{F}$-local t-spanners with $\mathcal{O}(n \log n)$ edges, when $\mathcal{F}$ is the family of all axis-parallel two-sided regions in the plane. To this end, we set G' to be an $\mathcal{F}$-local t-spanner for some $t < 2$ which therefore contains the closest pair for every possible query region.

Theorem 7. *A set S of n points in the plane can be stored in a structure of size $\mathcal{O}(n \log^5 n)$ such that for any axis-parallel query rectangle Q, the closest pair in $S \cap Q$ can be reported in $\mathcal{O}(\log^2 n)$ time. Moreover, the structure can be built in $\mathcal{O}(n \log^5 n)$ time.*

3.2.1. Local t-spanner.

In this section, we construct $\mathcal{F}$-local t-spanners with $\mathcal{O}(n \log n)$ edges, when $\mathcal{F}$ is the family of all axis-parallel two-sided regions in the plane. Due to similarity, we just consider the family $\mathcal{VS}$ of vertical strips and the family $\mathcal{NE}$ of north-east quadrants. Our construction is based on the region fault-tolerant t-spanner (Abam et al., 2009). To re-use the approach in Abam et al. (2009), we construct the graph such that the following property holds for every s-semi-separated pair (A, B) (assuming that $\text{radius}(D_A) \leq \text{radius}(D_B)$).

(I) For every region $R \in \mathcal{F}$ such that $R \cap A \neq \emptyset$ and $R \cap B \neq \emptyset$, the point in $R \cap B$ that is closest to the center of D_A is connected to a point in $R \cap A$.

This property is equivalent to Lemma 3.2 of Abam et al. (2009) and following a similar argument as in the proof of Lemma 3.3 of the same paper, proves that any graph satisfying this property is an $\mathcal{F}$-local t-spanner.

We will show how to satisfy property (I) using $\mathcal{O}(|A| + |B|)$ edges when regions are vertical strips and north-east quadrants. Therefore, this gives us an $\mathcal{F}$-local t-spanner that contains $\mathcal{O}(n \log n)$ edges.

Vertical strips. We first sort the points in A based on their x-coordinates. Then, for each point $b \in B$, we find two consecutive points $a, a' \in A$ surrounding b on the x-axis. We then connect b to both a and a'.

Lemma 8. *The above connecting schema satisfies property (I) and uses $\mathcal{O}(|A| + |B|)$ edges and can be performed in $\mathcal{O}((|A| + |B|) \log |A|)$ time.*

Proof. Assume that an arbitrary region $R \in \mathcal{VS}$ contains at least one point from each subset A and B. Let $a_1, \ldots, a_k$ be the sorted list of points in A, based on their x-coordinates. Let $b \in B \cap R$ be the point that is closest to the center of D_A. Our schema connects b to a_i and a_{i+1} for some i. If R does not contain a_i or a_{i+1}, then $R \cap A$ must be empty which is not true by assumption—note that since R is a vertical strip, it contains a contiguous subsequence of the sorted list. Therefore, the point b must be connected to a point of A.

Since the above schema just needs a sorted list of $|A|$, and it makes $|B|$ binary searches in this list, it can be performed in $\mathcal{O}((|A| + |B|) \log |A|)$ time. ■

North-east quadrants. For a point $p = (p_x, p_y)$, let NE(p) be the north-east quadrant with apex at p. More precisely, $\text{NE}(p) = [p_x, +\infty) \times [p_y, +\infty)$. Similarly we define $\text{NW}(p) = (-\infty, p_x] \times [p_y, +\infty)$ and $\text{SE}(p) = [p_x, +\infty) \times (-\infty, p_y)$. The connecting schema is as follows:

(1) We connect every point $a \in A$ to the point in $\text{NE}(a) \cap B$, if it exists, that is closest to the center of D_A.

(2) We connect each point $b \in B$ to an arbitrary point in $\text{NE}(b) \cap A$, to the highest point in $\text{SE}(b) \cap A$ and to the rightmost point in $\text{NW}(b) \cap A$, if they exist.

The following lemma shows our connecting schema holds the property (I).

Lemma 9. *The above connecting schema satisfies property (I) and uses $\mathcal{O}(|A| + |B|)$ edges and can be performed in $\mathcal{O}((|A| + |B|) \log^2(|A| + |B|))$ time.*

Proof. Let $R \in \mathcal{NE}$ be an arbitrary north-east quadrant containing at least one point of each subset A and B, and let $b \in R \cap B$ be the point that is closest to the center of D_A. If there exists a point of $R \cap A$ in NE(b), SE(b), or NW(b), our schema guarantees that b is connected to one of points in $R \cap A$. If this is not the case, then for every $a \in R \cap A$, the point b must be in NE(a) which then, by the first step of our schema, guarantees that a is connected to b.

To perform the above schema, we need two 2-dimensional range trees T_A and T_B for the points in A and B, respectively. We perform $|A|$ searches in T_B and $3|B|$ searches in T_A which in total can be done in $\mathcal{O}((|A| + |B|) \log^2(|A| + |B|))$ time. ■

4. SSPD Makes Life Easier

4.1. Spanners for complete k-Partite Graphs

Bose et al. (Bose et al., 2008) introduced the following problem: Given a complete k-partite graph K on a set of n points in $\mathbb{R}^d$, compute a sparse spanner of the graph K. They presented an algorithm running in $\mathcal{O}(n \log n)$ time that computes a $(5 + \varepsilon)$-spanner of K with $\mathcal{O}(n)$ edges. They also gave an algorithm of $\mathcal{O}(n \log n)$ time complexity that computes a $(3 + \varepsilon)$-spanner of K with $\mathcal{O}(n \log n)$ edges. This algorithm is based on a WSPD of the points and a bit involved. They also showed that every t-spanner of K for $t < 3$ must contain $\Omega(n \log n)$ edges.

We present a simpler algorithm, using the SSPD, to compute a $(3+\varepsilon)$-spanner of K with $\mathcal{O}(n \log n)$ edges in $\mathcal{O}(n \log n)$ time. We first present the algorithm when K is a complete bipartite graph; at the end, we describe how to extend it to any k-partite graph. To this end, assume that we are given a complete bipartite graph of n red and blue points.

We first compute an SSPD of the point set with respect to $s = 6/\varepsilon$, no matter what the color of the points is. Consider a pair (A, B) in the SSPD. There exist two disjoint balls D_A and D_B containing A and B, resp., such that $d(D_A, D_B) \geq s \cdot \min(\text{radius}(D_A), \text{radius}(D_B))$. Assume that $\text{radius}(D_A) \leq \text{radius}(D_B)$. We choose a red and a blue representative point in A, denoted by $\text{rep}_r(A)$ and $\text{rep}_b(A)$, resp., if they exist. We also choose red and blue representative points in B, denoted by $\text{rep}_r(B)$ and $\text{rep}_b(B)$, which are the red and the blue points in B that are closest to A. Then we connect $\text{rep}_r(A)$ to all blue points in B and $\text{rep}_b(A)$ to all red points in B. We apply the same procedure for the representative points in B.

Consider a pair (x, y) of points, where x is red and y is blue and assume that (A, B) is the pair in the SSPD such that $x \in A$ and $y \in B$. Assume that $\text{radius}(D_A) \leq \text{radius}(D_B)$. Our algorithm connects x to $\text{rep}_b(B)$, $\text{rep}_b(B)$ to $\text{rep}_r(A)$, and $\text{rep}_r(A)$ to y. Let Π be this 3-hop path between x and y. For ease of presentation let $z = \text{rep}_r(A)$ and $w = \text{rep}_b(B)$, and let o and r be the center and the radius of D_A. We have:

$$\begin{aligned}
|\Pi| &\leq |xw| + |wz| + |zy| \\
&\leq r + |wo| + r + |ow| + 2r + |xy| = 4r + 2|wo| + |xy| \\
&\leq 4r + 2|yo| + |xy| \\
&\leq 4r + 2(|xy| + r) + |xy| = 6r + 3|xy| \\
&\leq 6|xy|/s + 3|xy| = (3 + 6/s)|xy| = (3+\varepsilon)|xy|.
\end{aligned}$$

Extending the results to k-partite complete graphs is simple. We choose a representative point for any component for each color and we connect each representative to all the other points whose colors are different form the representative. This gives a $(3+\varepsilon)$-spanner of size $\mathcal{O}(kn \log n)$.

4.2. Low-Diameter Spanners

The diameter of a t-spanner is the minimum integer Δ such that for any pair of points, there exists a t-path between them in the t-spanner containing at most Δ links. Spanners with low diameter are desirable to many applications like ad hoc networks where in order to quickly get a packet to the receiver it must pass through few stations. There are several t-spanners with $\mathcal{O}(\log n)$ diameter and $\mathcal{O}(n)$ edges; for example see Arya et al. (1994, 1999); Bose et al. (2004). Moreover, Arya et al. (Arya et al., 1995) presented an algorithm for constructing a t-spanner of diameter 2 that contains $\mathcal{O}(n \log n)$ edges. They also showed that a t-spanner with a constant diameter cannot have a linear number of edges.

t-spanner with diameter 2. Computing t-spanner of diameter 2 is simple using the SSPD. We compute an SSPD of the points with respect to $4/(t-1)$. Then for each pair (A, B) in the SSPD, assuming $\text{radius}(D_A) \leq \text{radius}(D_B)$, we choose an arbitrary point p in A and connect all the points in $A \cup B \setminus \{p\}$ to p. This gives us a spanner with $\mathcal{O}(n \log n)$, because of the SSPD property.

Let p and q be two arbitrary points. There is a pair (A, B) in the SSPD such that $p \in A$ and $q \in B$ or vice-versa. Assume that $\text{radius}(D_A) \leq \text{radius}(D_B)$. Based on our construction, both p and q are connected to a point w in A. Since

$$|pw| + |wq| \leq 2|pw| + |pq| \leq 4\,\text{radius}(D_A) + |pq| \leq (t-1)|pq| + |pq| \leq t \cdot |pq|,$$

it follows that the spanner has diameter 2.

References

Abam, M. A., de Berg, M., Farshi, M., Gudmundsson, J., 2009. Region-fault tolerant geometric spanners. Discrete and Computational Geometry 41, 556–582.

Abam, M. A., de Berg, M., Farshi, M., Gudmundsson, J., Smid, M., Sep. 2011. Geometric spanners for weighted point sets. Algorithmica 61 (1), 207–225. URL `http://dx.doi.org/10.1007/s00453-010-9465-2`

Abam, M. A., Har-Peled, S., 2012. New constructions of SSPDs and their applications. Computational Geometry: Theory and Applications 45 (5-6), 200–214.
URL `http://dx.doi.org/10.1016/j.comgeo.2011.12.003`

Agarwal, P. K., Erickson, J., 1999. Geometric range searching and its relatives. In: Advances in Discrete and Computational Geometry. American Mathematical Society, pp. 1–56.

Arya, S., Das, G., Mount, D. M., Salowe, J. S., Smid, M., 1995. Euclidean spanners: short, thin, and lanky. In: STOC'95: Proceedings of the 27th Annual ACM Symposium on Theory of Computing. pp. 489–498.

Arya, S., Mount, D. M., Smid, M., 1994. Randomized and deterministic algorithms for geometric spanners of small diameter. In: FOCS'94: Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science. pp. 703–712.

Arya, S., Mount, D. M., Smid, M., 1999. Dynamic algorithms for geometric spanners of small diameter: Randomized solutions. Computational Geometry: Theory and Applications 13 (2), 91–107.

Bollobás, B., Scott, A., 2007. On separating systems. European Journal of Combinatorics 28 (4), 1068–1071.

Bose, P., Carmi, P., Courture, M., Maheshvari, A., Morin, P., Smid, M., 2008. Spanners of complete k-partite geometric graphs. In: LATIN'08: Proceedings of the 8th Latin American Theoretical Informatics Symposium . Vol. 4957 of Lecture Notes in Computer Science. Springer, pp. 170–181.

Bose, P., Gudmundsson, J., Morin, P., 2004. Ordered theta graphs. Computational Geometry: Theory and Applications 28, 11–18.

Callahan, P. B., Kosaraju, S. R., 1993. Faster algorithms for some geometric graph problems in higher dimensions. In: SODA'93: Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 291–300.

Callahan, P. B., Kosaraju, S. R., 1995. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. Journal of the ACM 42, 67–90.

Chan, T. M., 2012. Optimal partition trees. Discrete and Computational Geometry 47 (4), 661–690.

de Berg, M., Cheong, O., van Kreveld, M., Overmars, M., 2008. Computational Geometry: Algorithms and Applications, 3rd Edition. Springer-Verlag, Berlin, Germany.

Duncan, C. A., Goodrich, M. T., Kobourov, S., 2001. Balanced aspect ratio trees: Combining the advances of k-d trees and octrees. Journal of Algorithms 38, 303–333.

Edelsbrunner, H., Mücke, E. P., 1990. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Transactions on Graphics 9, 66–104.

Eppstein, D., 2000. Spanning trees and spanners. In: Sack, J.-R., Urrutia, J. (Eds.), Handbook of Computational Geometry. Elsevier Science Publishers, Amsterdam, pp. 425–461.

Gudmundsson, J., Knauer, C., 2007. Dilation and detour in geometric networks. In: Gonzalez, T. (Ed.), Handbook on approximation algorithms and metaheuristics. Chapman & Hall/CRC, Amsterdam, pp. 52–1–52–17.

Gupta, P., 2006. Range-aggregate query problems involving geometric aggregation operations. Nordic Journal of Computing 13 (4), 294–308.

Gupta, P., Janardan, R., Kumar, Y., Smid, M., 2008. Data structures for range-aggregate extent queries. In: CCCG'08: Proceedings of the 20th Canadian Conference on Computational Geometry. pp. 7–10.

Hansel, G., 1964. Nombre minimal de contacts de fermeture nécessaires pour réaliser une fonction booléenne symétrique de n variables. C. R. Acad. Sci. Paris 258, 6037–6040, russian transl., Kibern. Sb. (Nov. Ser.) 5 (1968), 47-52.

Löffler, M., Snoeyink, J., 2008. Delaunay triangulations of imprecise points in linear time after preprocessing. In: SCG'08: Proceedings of the 24th Annual ACM Symposium on Computational Geometry. ACM, New York, NY, USA, pp. 298–304.

Narasimhan, G., Smid, M., 2007. Geometric spanner networks. Cambridge University Press.

Nievergelt, J., Widmayer, P., 2000. Spatial data structures: Concepts and design choices. In: Sack, J.-R., Urrutia, J. (Eds.), Handbook of Computational Geometry. Elsevier Science Publishers, Amsterdam, pp. 725–764.

Shan, J., Zhang, D., Salzberg, B., 2003. On spatial-range closest-pair query. In: In Proceedings of Symposium on Advances in Spatial and Temporal Databases (SSTD). Vol. 2750 of Lecture Notes in Computer Science. Springer Verlag, pp. 252–269.

Sharathkumar, R., Gupta, P., 2007. Range-aggregate proximity queries. Technical Report IIIT/TR/2007/80, International Institute of Information Technology Hyderabad.
URL `http://www.iiit.ac.in/techreports/2007_80.pdf`

Smid, M., 2000. Closest point problems in computational geometry. In: Sack, J.-R., Urrutia, J. (Eds.), Handbook of Computational Geometry. Elsevier Science Publishers, Amsterdam, pp. 877–935.

Tao, Y., Papadias, D., December 2004. Range aggregate processing in spatial databases. IEEE Transactions on Knowledge and Data Engineering 16 (12), 1555–1570.

van Kreveld, M., Löffler, M., 2008. Approximating largest convex hulls for imprecise points. Journal of Discrete Algorithms 6 (4), 583–594.

Varadarajan, K. R., 1998. A divide-and-conquer algorithm for min-cost perfect matching in the plane. In: FOCS'98: Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science. pp. 320–331.