Closest Pair and the Post Office Problem for Stochastic Points*

Pegah Kamousi† Timothy M. Chan‡ Subhash Suri§

Abstract

Given a (master) set M of n points in d-dimensional Euclidean space, consider drawing a random subset that includes each point $m_i \in M$ with an independent probability p_i. How difficult is it to compute elementary statistics about the closest pair of points in such a subset? For instance, what is the *probability* that the distance between the closest pair of points in the random subset is no more than ℓ, for a given value ℓ? Or, can we preprocess the master set M such that given a query point q, we can efficiently estimate the *expected* distance from q to its nearest neighbor in the random subset? These basic computational geometry problems, whose complexity is quite well-understood in the deterministic setting, prove to be surprisingly hard in our *stochastic* setting. We obtain hardness results and approximation algorithms for stochastic problems of this kind.

1 Introduction

Many years ago, Knuth [12] posed the now classic *post-office* problem, namely, given a set of points in the plane, find the one closest to a query point q. The problem, which is fundamental and arises as a basic building block of numerous computational geometry algorithms and data structures [7], is reasonably well-understood in small dimensions. In this paper, we consider a *stochastic* version of the problem in which each post office may be *closed* with certain probability. In other words, a given set of points M in d dimensions includes the locations of all the post offices but on a typical day each post office $m_i \in M$ is only open with an independent probability p_i. The set of points M together with their probabilities form a probability distribution where each point m_i is included in a random subset of points with probability p_i. Thus, given a query points q, we now must talk about the *expected* distance from q to its closest neighbor in M. Similarly, instead of simply computing the closest pair of points in a set, we must ask: how *likely* is it that the closest pair of points are no more than ℓ apart?

In this paper, we study the complexity of such elementary proximity problems and establish both upper and lower bounds. In particular, we have a finite set of points M in a d-dimensional Euclidean space, which constitutes our *master* set of points, and hence the mnemonic M. Each member m_i of M has probability p_i of being present and probability $1 - p_i$ of being absent. (Following the post-office analogy, the ith post office is open with probability p_i and closed otherwise.) These

*The work of the first and the third author was supported in part by National Science Foundation grants CCF-0514738 and CNS-1035917. The work of the second author was supported by NSERC. A preliminary version of the paper has appeared in *Proc. 12th Algorithms and Data Structures Symposium* (WADS), pages 548–559, 2011.

†Department of Computer Science, UC Santa Barbara, CA 93106, USA.

‡Cheriton School of Computer Science, University of Waterloo, Ontario N2L 3G1, Canada.

§Department of Computer Science, UC Santa Barbara, CA 93106, USA.

probabilities are independent, but otherwise arbitrary, and lead to a sample space of 2^n possible subsets, where a sample subset includes the ith point with independent probability p_i. We achieve the following results.

1. It is #P-hard to compute the probability that the closest pair of points have distance at most a value ℓ, even for dimension 2 under the L_∞ norm.

2. In the *linearly-separable* and *bichromatic* planar case, the above closest pair probability can be computed in polynomial time under the L_∞ norm.

3. Without the linear separability, even the bichromatic version of the above problem is #P-hard under the L_2 or L_∞ norm.

4. Even with linear separability and L_∞ norm, the bichromatic case becomes #P-hard in dimension $d \geq 3$.

5. We give a linear-space data structure with $O(\log n)$ query time to compute the expected distance of a given query point to its $(1+\varepsilon)$-approximate nearest neighbor when the dimension d is a constant.

Related Work

A number of researchers have recently begun to explore geometric computing over *probabilistic* data [1, 2, 14, 19]. These studies are fundamentally different from the classical geometric probability that deals with properties of random point sets drawn from some infinite sets, such as points in unit square [11]. Instead, the recent work in computational geometry is concerned with worst-case sets of objects and worst-case probabilities or behavior. In particular, the recent works in [1, 2] deals with the database problem of skyline computation using a multiple-universe model. The work of van Kreveld and Löffler [14, 19] deals with objects whose locations are randomly distributed in some *uncertainty* regions. Guibas et. al. [8] consider the behavior of fundamental geometric constructs such as the Voronoi diagrams when the point generators are not known precisely but only given as Gaussian distributions. Unlike these results, our focus is not on the locational uncertainty but rather on the discrete probabilities with which each point may appear.

2 The Stochastic Closest Pair Problem

We begin with the complexity of the stochastic closest pair (SCP) problem in the plane, which asks for the probability that the closest pair of points in the plane has distance at most a given bound ℓ. We use the notation $d(P, Q)$ for the L_2 distance between two sets P and Q. (When using the L_∞ or L_1 norms, we will use $d_\infty(P, Q)$ and $d_1(P, Q)$.) Consider a set of points $M = \{m_1, m_2, \ldots, m_n\}$ in the plane, called the *master* set. Each point m_i is *active*, or present, with some independent and arbitrary but known (rational-valued) probability p_i. The independent point probabilities induce a sample space Ω with 2^n outcomes. Let $S \subset M$ denote the set of active points in a trial. Then an outcome $A \subseteq M$ occurs with probability $\Pr[S = A] = \prod_{m_i \in A} p_i \prod_{m_i \notin A} (1 - p_i)$. We ask for the probability that $\min_{m_i, m_j \in S} d(m_i, m_j) \leq \ell$. We will show that this basic problem is intractable, via reduction from the problem of *counting minimum vertex covers* in planar unit disk graphs. In order

to show that even the *bichromatic* closest pair problem is hard, we also prove that a corresponding vertex cover counting problem is also hard for a bichromatic version of the unit disk graphs.

2.1 Counting Vertex Covers in Unit Disk Graphs

The problem of counting minimum cardinality vertex covers in a graph [13, 16, 18] is the following. Given a graph $G = (V, E)$, how many node-subsets $S \subseteq V$ constitute a vertex cover of minimum cardinality, where S is a vertex cover of G if for each $uv \in E$, either $u \in S$ or $v \in S$. This problem is known to be #P-hard, even for planar bipartite graphs with maximum degree 3 [16]. This immediately shows that counting the minimum *weight* vertex covers is also hard for this type of graphs, as we can always assign the same weight to all the node, so that the minimum cardinality vertex cover minimizes the weight as well. Throughout this paper, we simply use the term *minimum vertex cover* instead of minimum *weight* vertex cover, and the term minimum *cardinality* vertex cover is used otherwise.

Let **#MinVC** denote the problem of counting minimum weight vertex covers in a graph. We will consider this problem on *unit disk graphs*, which are the intersection graphs of n equal-sized circles in the plane: each node corresponds to a circle, and there is an edge between two nodes if the corresponding circles intersect. The following theorem considers the complexity of **#MinVC** problem on unit disk graphs.

Theorem 2.1. *It is #P-hard to compute the number of minimum weight vertex covers in weighted unit disk graphs, even if all the nodes have weight* 1 *or* 2*, and even if the distances are measured in the L_∞ metric.*

Throughout this section, we assume that the disks defining unit disk graphs have radius 1. The first step of the proof is the following well-known lemma of Valiant [17].

Lemma 2.2. *A planar graph $G = (V, E)$ with maximum degree* 4 *can be embedded in the plane using $O(|V|^2)$ area in such a way that its nodes are at integer coordinates and its edges are drawn so that they are made up of line segments of the form $x = i$ or $y = j$, for integers i and j.*

Our reduction is from the problem of counting minimum cardinality vertex covers in planar bipartite graphs with maximum degree 3, which is #P-hard as proved by Vadhan [16]. Let $G = (V, E)$ be an instance of this problem and let m and n denote the number of edges and nodes of G, respectively.

Let P be a planar embedding of G according to Lemma 2.2, and G_i be the graph obtained by putting an even number $2L_i$ of *intermediate* nodes on each edge of P, for a given parameter L_i (see Figure 1). Finally we assign weight 2 to all the intermediate nodes, and weight 1 to *regular* nodes (the images of the nodes in G). We also assign weight 1 to all the nodes in G, so that a minimum weight vertex cover of G also has minimum cardinality. The following lemma relates the weight of minimum vertex covers in G and G_i.

Lemma 2.3. *G has a minimum vertex cover of weight w if and only if G_i has a minimum vertex cover of weight $w + 2mL_i$.*

Proof. For any edge $uv \in E$, let $path(u, v)$ denote the path connecting the image of u to the image of v in G_i. For each minimum vertex cover C in G with weight w, it can be extended to a vertex cover of weight $w + 2\sum_{uv \in E} L_i = w + 2mL_i$ in G_i by including every other intermediate node on $path(u, v)$ for each edge $uv \in E$.

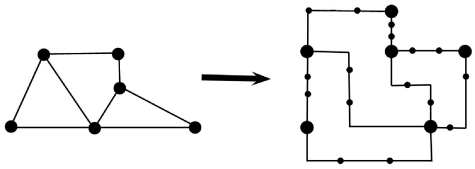

Figure 1: A planar grid embedding of a graph with 2 intermediate nodes on each edge.

On the other hand, every minimum vertex cover C_i of G_i is also a minimum vertex cover for G, when restricted to the regular nodes. Suppose not. Then there exists at least one path, $path(u, v)$, such that neither u nor v belongs to C_i. Consider the path in Figure 2(a). If u and v are not selected (the first scenario), both a and b should be selected to cover the edges ua and bv. Half of the remaining intermediate nodes are necessary and sufficient to cover the remaining edges (since for any edge on the path at least one of its endpoints must belong to the vertex cover). The weight of the nodes covering this path is therefore $2L_i + 2$. By including the node u and taking out a, we still have a vertex cover whose weight is now reduced to $2L_i + 1$ (Figure 2(a), second scenario). But this is a contradiction since C_i is a minimum weight vertex cover.

This shows that any minimum vertex cover for G_i contains a set of regular nodes which is a vertex cover for G, plus L_i intermediate nodes per each edge of G. The weight of C_i is therefore minimized if the set of regular nodes in C_i is a minimum vertex cover for G. □

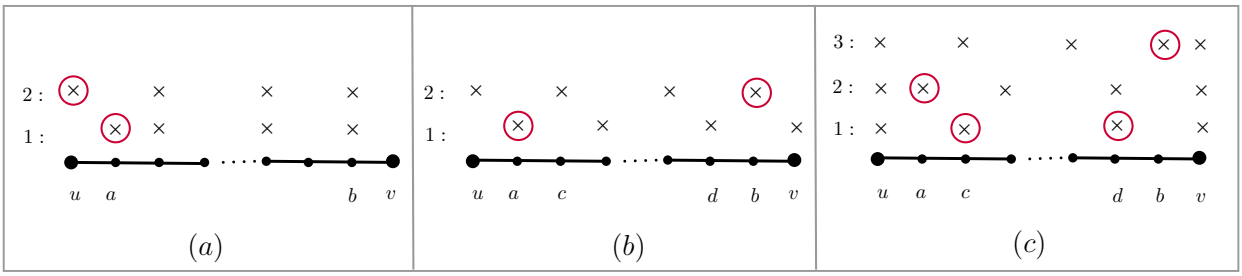

Figure 2: Counting minimum vertex covers in G_i.

Let $N(G)$ and $N(G_i)$ denote the number of minimum vertex covers of G and G_i respectively. For $j = 0, \ldots, m$, define N_j as the number of minimum vertex covers of G in which exactly j edges of G have both (the images of) their endpoint covered, and $m - j$ edges have one of their endpoints covered. The following lemma relates the number of minimum vertex covers in G and G_i.

Lemma 2.4. *Let G and G_i be the graphs defined above. Then we have*

$$N(G_i) = \sum_{j=0}^{m} N_j (L_i + 1)^j, \tag{1}$$

and the number of vertex covers of G is simply $N(G) = \sum_{j=0}^{m} N_j$.

Proof. Consider the path in Figure 2(b), which is a path in G_i corresponding to an edge uv from G plus $2L_i$ intermediate nodes.

Let C_i be a minimum vertex cover for G_i. If only one of u and v belongs to C_i as in Figure 2(b), then the only way to cover the remaining edges with minimum weight is to include every other intermediate node in the vertex cover, starting from the selected node. Now consider the case where both u and v belong to C_i as in Figure 2(c). The claim is that in this case there are exactly $L_i + 1$ different ways to minimally cover the remaining edges. This is true for $L_i = 1$, where we have two intermediate nodes each of which can be selected, and it is easy to see that it also holds for $L_i = 2$. Now suppose this is true for $L_i \leq k$ and consider the case where there are $2(k+1)$ intermediate nodes on the edge. See Figure 2(c). If neither a nor b is selected (first scenario), both c and d need to be selected. Then the path connecting c to d has both its endpoints covered, and we are back to the case where there are $2(k-1)$ intermediate nodes, and there are exactly k ways to minimally cover the remaining edges. If only a is selected (second scenario), then the path connecting a to b has only one of its endpoints covered, and therefore there is only one way to cover the remaining edges. The same happens if only b is selected (the third scenario).

Therefore the number of ways to minimally cover $path(u, v)$, given that both its endpoints are selected, is $k + 2 = (k + 1) + 1$. This completes the induction. We conclude that there are $(L_i + 1)^j$ different minimum vertex covers of G_i corresponding to each minimum vertex cover of G, in which j edges of G have both their endpoint covered. This completes the proof. □

What remains is to recover the values N_j from the value of $N(G_i)$. We will need the following lemma from [15] (see Section 2).

Lemma 2.5. *Suppose we have $v_0, \ldots, v_n, b_0 \ldots, b_n$, related by the equation $v_i = \sum_{j=0}^{n} a_{ij} b_j$ $(i = 0, \ldots, n)$. Further suppose that the matrix of the coefficients (a_{ij}) is Vandermonde, with parameters $\mu_0, \ldots, \mu_n$ which are distinct. Then given the values $v_0, \ldots, v_n$, we can obtain the values $b_0, \ldots, b_n$ in time polynomial in n.*

Recall that a *Vandermonde matrix* $\mathbf{M}$ is in the form $\mathbf{M_{ij}} = (\mu_i^j)_{0 \leq i,j \leq n}$ (or its transpose) for a given sequence of distinct values $\mu_0, \ldots, \mu_n$. (See [9, §5.1].)

If we compute the sum $\sum_{j=0}^{m} N_j (L_i + 1)^j$ for $m + 1$ different values of L_i, we end up with $m + 1$ equations like equation (1) in Lemma 2.4, involving $m + 1$ unknowns. The matrix of coefficients, (a_{ij}) is Vandermonde, where $a_{ij} = (L_i + 1)^j$. We can then use Lemma 2.5 to recover the values N_j for $j = 0, \ldots, m$, and therefore the number of minimum vertex covers of G.

So far we know how to compute the number of minimum vertex covers of G in polynomial time, if we can count the minimum weight vertex covers of G_i. But we are still missing the main piece. In order to prove Theorem 2.1, the G_i's need to be unit disk graphs. The rest of this section will show how to embed the G_i's as unit disk graphs in the plane for appropriate values of L_i.

Let P, as defined before, be a planar embedding of G in the integer plane grid according to Lemma 2.2. Let $M \in O(n^2)$ be an integer upper bound on the length of the longest edge of P, with $M \geq m$. First we scale the grid by the factor $80M$ (the length of the longest edge is now bounded by $80M^2$). Next we place a number of intermediate nodes on all the edges, such that each edge becomes a path consisting of edges of length 1 (*short edges* hereafter). This is always possible since even after scaling the grid all the edges have integer lengths.

As before, for each edge $uv \in E$, $path(u, v)$ denotes the path connecting the image of u to the image of v in the resulting graph. For any given $i = 0, \ldots, m$, we transform each of these m paths into a path consisting of exactly $\ell_i = 80M^2 + 40iM$ short edges as follows: After the scaling, all the paths have length at least $80M$. On each path, we take a vertical or horizontal segment of length $4j$ with $j \leq 5M$, and replace it with the path in Figure 3 (the figure depicts the case of a

vertical segment being replaced; the horizontal case is symmetric). These paths are called (vertical or horizontal) *equalizing* paths, each of which fits inside a $20M \times 20M$ square. On each path, we can select a horizontal or vertical segment that is at distance more than $20M + 1$ away from any node or corner of the path; this ensures that the equalizing paths are more than distance 1 apart from one another.

Each vertical equalizing path consists of $2j$ horizontal segments of length $20M$ and $2j$ vertical segments of length 2. The length of the whole path is therefore $40jM + 4j$; in other words, the increase in length is $40jM$. By varying j from 0 to $5M$, this increase can assume any multiple of $40M$ between 0 to $200M^2$, and therefore it can be adjusted so that the length of $path(u, v)$ is exactly ℓ_i.

One remaining problem is that although all paths $path(u, v)$ now have the same number $\ell_i - 1$ of intermediate nodes, this number is odd. We resolve this problem by picking any line segment of length 2 in $path(u, v)$ and subdividing into 3 edges of length 2/3 (instead of 2 edges of length 1). The resulting unit disk graph will contain short edges of length 1 and these new edges of length 2/3, but no other edges. Now for each $uv \in E$, $path(u, v)$ have $2L_i$ intermediate nodes with $L_i = \ell_i/2$.

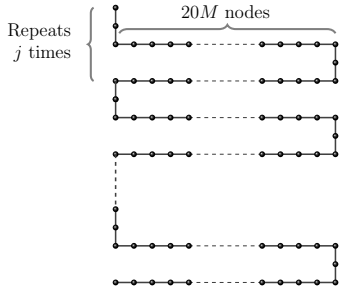

Figure 3: A vertical equalizing path.

We are now ready to prove Theorem 2.1.

Proof of Theorem 2.1. Given $G = (V, E)$, a bipartite planar graph of maximum degree 3, we construct a series of $m + 1$ unit disk graphs G_i, $i = 0, \ldots, m$, where each G_i has exactly $2L_i = 80M^2 + 40iM$ intermediate nodes on $path(u, v)$, for all $uv \in E$, and M is defined as before. We assign weight 1 to all the nodes in G and their images in the G_i's, while we assing weight 2 to all the intermediate nodes of the G_i's. Suppose we have an algorithm to count the minimum vertex covers in unit disk graphs. We then use this algorithm to count the minimum vertex covers in all the G_i's, and use Lemma 2.5 along with Lemma 2.4 to recover the number of minimum weight (and therefore cardinality) vertex covers of G. But counting the minimum cardinality vertex covers is hard for bipartite graphs of maximum degree 3, which shows that counting the minimum weight vertex covers is hard for unit disk graphs. This holds even for planar, bipartite unit disk graphs since the G_i's are bipartite and planar. The hardness result also holds if the distances are measured in the L_∞ metric, since the G_i's only have horizontal and vertical segments, and therefore the distances do not change under the L_∞ metric. □

In the next section we introduce the class of *Bichromatic Unit Disk Graphs*, and show that counting the vertex covers remains hard in this version of unit this graphs. This result will be needed later to prove that the SCP problem is hard even for bichromatic point sets.

2.2 Bichromatic Unit Disk Graphs

We introduce the class of *bichromatic unit disk graphs* as the graphs defined over a set of points in the plane, each colored as blue or red, with an edge between a red and a blue pair if and only if their distance is ≤ 1. (We do not put edges between nodes of the same color regardless of the distance between them.) The next theorem shows that counting minimum vertex covers remains #P-hard for bichromatic UDGs.

Theorem 2.6. *It is #P-hard to count the minimum weight vertex covers in a bichromatic unit disk graph even if all the nodes have weight* 1 *or* 2*, and even if the distances are measured in the* L_∞ *metric.*

Proof. Consider the graphs G_i for $i = 0, \ldots, m$ constructed in the proof of Theorem 2.1. These graphs are bipartite, and we can therefore color their nodes as blue and red such that there is no edge between a red and a blue node. We can now apply the exact same reduction as in the proof of Theorem 2.1, to show that counting the minimum weight vertex covers is #P-hard for bichromatic unit disk graphs as well. □

In the next section we study the close connection between the problem of counting the minimum vertex covers in unit disk graphs and the stochastic closest pair problem.

2.3 Complexity of the Stochastic Closest Pair Problem

Let $M = \{m_1, \ldots, m_n\}$ be a set of n points in the plane, where each $m_i \in M$ is present with probability p_i and absent with probability $1 - p_i$. Let H be the (stochastic) unit disk graph defined on M, with an edge between two points if and only if their distance is ≤ 1. In this graph, a subset S of nodes is a vertex cover if and only if no two nodes in the complement of S are at distance ≤ 1. (In other words, all the edges are covered by S.) Therefore, computing the probability that a random subset of nodes is a vertex cover in H amounts to computing the probability that the closest pair of the complement of the random subset are at distance > 1.

With this intuition, we are now ready to prove the main result of this section.

Theorem 2.7. *Given a set M of points in the plane, where each point $m_i \in M$ is present with probability p_i, it is #P-hard to compute the probability that the L_2 or L_∞ distance between the closest pair is $\leq \ell$, for a given value ℓ.*

Proof. The reduction is from the **#MinVC** problem for unit disk graphs in which all the nodes have weight 1 or 2. Consider such a unit disk graph H. We assign probability $1 - q$ to all the nodes of weight 1, and probability $1 - q^2$ to all the nodes of weight 2, for a value q to be determined later. Let n_1 and n_2 be the number of nodes of weight 1 and 2 in H respectively.

Suppose there is a polynomial time algorithm for the SCP problem. We run that algorithm on the nodes of H, to get the probability that the closest pair of the random subset are at distance > 1, in other words, the probability that the complement of the random subset is a vertex cover. Let $P(H)$ denote this probability.

The probability that the complement of a random subset equals a fixed subset with j_1 nodes of weight 1, j_2 nodes of weight 2, and total weight $j = j_1 + 2j_2$ is $q^{j_1}(q^2)^{j_2}(1-q)^{n_1-j_1}(1-q^2)^{n_2-j_2} \in q^j(1 - O(nq))$. Thus,

$$P(H) \in \sum_{j=1}^{2n} C_j \cdot q^j(1 - O(nq)),$$

where C_j is the number of vertex covers of weight j.

Let k be the weight of the minimum vertex cover. Since the total number of subsets is 2^n,

$$P(H) \in C_k \cdot q^k(1 - O(nq)) + O(2^n \cdot q^{k+1}(1 - O(nq))) \subset (C_k \pm o(1)) \cdot q^k$$

by choosing a sufficiently small value of $q \ll 1/n2^n$. (Note that we can still make q have polynomial number of bits.) We can find k by searching for the closest power of q to $P(H)$. We can divide $P(H)$ by q^k and round to the nearest integer to get C_k, the number of minimum weight vertex covers in H. But Theorem 2.1 asserts that this problem is #P-hard, which shows that the SCP problem is #P-hard as well. □

The next theorem, which considers the bichromatic version of this problem, is based on the same argument as above along with Theorem 2.6.

Theorem 2.8. *Given a set R of red and a set B of blue points in the plane, where each point $r_i \in R$ is present with an independent, rational probability p_i, and each point $b_i \in B$ is present with probability q_i, it is #P-hard to compute the probability that the closest L_2 or L_∞ distance between a bichromatic pair of present points is less that a given value ℓ.*

In the next section, we propose a polynomial time algorithm for a special case of the bichromatic closest pair problem.

3 Linearly Separable Point Sets under the L_∞ Norm

In the following, we show that when the red points are linearly separable from the blue points by a vertical or a horizontal line, the stochastic bichromatic closest pair problem under L_∞ distances can be solved in polynomial time. We only describe the algorithm for the points separable by a vertical line, noting that all the arguments can be adapted to the case of a horizontal line.

Let $U = \{u_1, \ldots, u_n\}$ be the set of red points on the left side, and $V = \{v_1, \ldots, v_m\}$ the set of blue points on the right side of a line. Each point $u_i \in U$ is present with probability p_i, while each point $v_j \in V$ is present with probability q_j. We sort the red points by x-coordinate (from right to left), and the blue points by y-coordinate (top-down). Let $R[i, j, k]$ be the region defined by the intersection of the halfplanes $x \leq 0, x \geq x(u_i) - 1, y \geq y(v_j)$ and $y \leq y(v_k)$, for $y(v_j) < y(v_k)$ (Fig. 4 (a), where $x(u_i)$ and $y(v_j)$ denote the x-coordinate of the i-th red point and the y-coordinate of the j-th blue point, respectively. By abuse of notation, we will also use $R[i, j, k]$ to refer to the set of (blue) points inside this region.

Let $P[i, j, k]$ denote the probability that the subset $U_i = \{u_1, u_2, \ldots, u_i\}$ of red points does **not** have a neighbor within distance ≤ 1 in $R[i, j, k]$. The value we are interested in is $P[n, m, 1]$, which is the probability that the closest pair distance is > 1. We fill up the table for $P[i, j, k]$ values using dynamic programming, row by row starting from u_1 (the rightmost red point).

Let $B(u_i)$ be the L_∞ ball of radius 1 around u_i. In the computation of the entry $P[i, j, k]$, there are 4 cases:

1. $B(u_i)$ contains the region $R[i, j, k]$ (Fig. 4 (a)). In this case

$$P[i, j, k] = p_i \prod_{v_t \in B(u_i)} (1 - q_t) + (1 - p_i) \cdot P[i - 1, j, k].$$

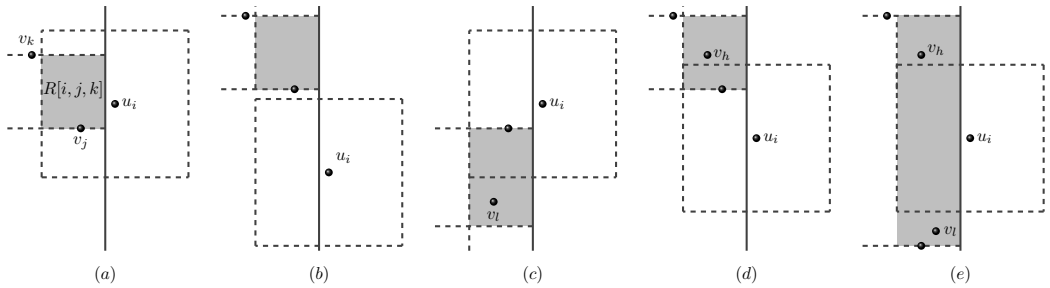

Figure 4: Different configurations of $R[i,j,k]$ and $B(u_i)$.

2. $B(u_i)$ does not intersect with $R[i,j,k]$ (Fig. 4 (b)). In this case, $P[i,j,k] = P[i-1,j,k]$.

3. The left half of $B(u_i)$ partially intersects with $R[i,j,k]$. If $y(u_i) - 1 < y(v_k)$ (Fig. 4 (c)),

$$P[i,j,k] = p_i \prod_{v_t \in B(u_i) \cap R[i,j,k]} (1-q_t) \cdot P[i-1,j,l] + (1-p_i) \cdot P[i-1,j,k],$$

where v_l is the highest blue point in $R[i,j,k]$ but outside $B(u_i)$.

If $y(u_i) + 1 < y(v_k)$) (Fig. 4 (d)), then

$$P[i,j,k] = p_i \prod_{v_t \in B(u_i) \cap R[i,j,k]} (1-q_t) \cdot P[i-1,h,k] + (1-p_i) \cdot P[i-1,j,k],$$

where v_h is the lowest blue point in $R[i,j,k]$ but outside $B(u_i)$.

4. The left half of $B(u_i)$ is contained in $R[i,j,k]$ (Fig. 4 (e)). In this case

$$P[i,j,k] = (1-p_i) \cdot P[i-1,j,k] + p_i \prod_{v_t \in B(u_i) \cap R[i,j,k]} (1-q_t) \cdot P[i-1,j,l] \cdot P[i-1,h,k],$$

where v_l and v_h are defined as before. The subtlety is that the two events of U_{i-1} having no close neighbor in $R[i-1,j,l]$, and of U_{i-1} having no close neighbor in $R[i-1,h,k]$ are independent. Therefore we can multiply the corresponding probabilities. The reason is that all the points in U_{i-1} that potentially have a close neighbor in $R[i-1,j,l]$ must necessarily lie below the line $y = y(u_i)$, while those potentially close to a point in $R[i-1,h,k]$ must lie above that line. The two sets are therefore disjoint.

The base case ($R[1,j,k], j,k \in \{1,\ldots,m\}$) can be easily computed. The size of the table is $O(n^3)$. The values $\prod_{v_t \in B(u_i) \cap R[i,j,k]}(1-q_t)$ can be precomputed in $O(n^2)$ time for each point v_i (by a sweep-line approach for example). This brings the computation time of each table entry down to a constant, and the running time of the whole algorithm to $O(n^3)$ time. This assumes a nonstandard RAM model of computation where each arithmetic operation on large numbers takes unit time. Otherwise, the running time should be multiplied by a factor proportional to the bit complexity of the intermediate numbers, which is polynomial in n and the bit complexity of the input probability values.

The next Theorem considers the problem for $d > 2$.

Theorem 3.1. *Given a set R of red and a set B of blue points in a Euclidean space of dimension $d > 2$, each being present with an independent, rational probability, it is #P-hard to compute the probability that the L_∞ distance between the closest pair of bichromatic points is less than a given value r, even when the two sets are linearly separable by a hyperplane orthogonal to some axis.*

Proof. Let $d_\infty(R, B)$ be the minimum L_∞ distance between all the bichromatic pairs, which lie in the plane. It is always possible to make the two sets linearly separable in $d = 3$ by lifting all the blue (or red) points from the plane by a small value $\epsilon < d_\infty(R, B)$. This does not change the L_∞ distance of any pair of points. Therefore, an algorithm for solving the problem for linearly separable sets in $d > 2$, is essentially an algorithm for the stochastic bichromatic closest pair problem, which is #P-hard by Theorem 2.8. This completes the proof. □

In the next section, we consider the stochastic version of the nearest neighbor search.

4 Stochastic Approximate Nearest Neighbor Queries

Given a stochastic set M of points in a d-dimensional Euclidean space, and a query point q, what is the expected (L_2) distance of q to the closest present point of M? In this section we target this problem, and design a data structure for approximating the expected value of $d(S, q) = \min_{p \in S} d(p, q)$ with respect to a random subset S of M, assuming that d is a constant. (Technically, at least one point needs to be assigned probability 1 to ensure that the expected value is finite; alternatively, we can consider the expectation conditioned on the event that $d(S, q)$ is upper-bounded by a fixed value.) We obtain a linear-space data structure with $O(\log n)$ query time. Although our method is based on known techniques for approximate nearest neighbor search (namely, balanced quadtrees and shifting [3, 4, 5]), a nontrivial adaptation of these techniques is required to solve the stochastic version of the problem.

4.1 Approximation via a modified distance function $\widetilde{\ell}$

As before, we are given a set M of points in a d-dimensional Euclidean space and each point is present with an independent probability. Assume that the points lie in the universe $\{0, \ldots, 2^w - 1\}^d$. Fix an odd integer $k = \Theta(1/\varepsilon)$. Shift all points in M by the vector $(j2^w/k, j2^w/k, \ldots, j2^w/k)$ for a randomly chosen $j \in \{0, \ldots, k-1\}$.

A *quadtree box* is a box of the form $[i_1 2^\ell, (i_1+1)2^\ell) \times \cdots \times [i_d 2^\ell, (i_d+1)2^\ell)$ for natural numbers $\ell, i_1, \ldots, i_d$. Given points p and q, let $\mathcal{D}(p, q)$ be the side length of the smallest quadtree box containing p and q. Let $B_s(p)$ be the quadtree box of side length $\lfloor\!\lfloor s \rfloor\!\rfloor$ containing p, where $\lfloor\!\lfloor s \rfloor\!\rfloor$ denotes the largest power of 2 smaller than s. Let $c_s(p)$ denote the center of $B_s(p)$. Let $[X]$ be 1 if X is true, and 0 otherwise.

Definition 4.1.

(a) *Define $\ell(p, q) = d(B_s(p), B_s(q)) + 2\sqrt{d}s$ with $s = \varepsilon^2 \mathcal{D}(p, q)$. Let $\ell(S, q) = \min_{p \in S} \ell(p, q)$.*

(b) *r is said to be* q-good *if the ball centered at $c_{\varepsilon^2 r}(q)$ of radius $2r$ is contained in $B_{12kr}(q)$.*

(c) *Define $\widetilde{\ell}(S, q) = [\ell(S, q) \text{ is } q\text{-good}] \cdot \ell(S, q)$.*

Lemma 4.2.

(a) $\ell(S,q) \geq d(S,q)$. *Furthermore, if* $\ell(S,q)$ *is* q*-good, then* $\ell(S,q) \leq (1+O(\varepsilon))d(S,q)$.

(b) $\ell(S,q)$ *is* q*-good for all but at most* d *choices of the random index* j.

(c) $\widetilde{\ell}(S,q) \leq (1+O(\varepsilon))d(S,q)$ *always, and* $\mathbb{E}_j[\widetilde{\ell}(S,q)] \geq (1-O(\varepsilon))d(S,q)$.

Proof. Let $p^*, p \in S$ satisfy $d(S,q) = d(p^*,q) = r^*$ and $\ell(S,q) = \ell(p,q) = r$.

The first part of (a) follows since $\ell(p,q) \geq d(p,q)$. For the second part of (a), suppose that r is q-good. Since $d(p^*,q) \leq d(p,q) \leq \ell(p,q) = r$, we have $d(p^*, c_{\varepsilon^2 r}(q)) < 2r$, implying $\mathcal{D}(p^*,q) \leq 12kr$. Then $r = \ell(p,q) \leq \ell(p^*,q) \leq d(p^*,q) + O(\varepsilon^2 \mathcal{D}(p^*,q)) \leq r^* + O(\varepsilon r)$, and so $r \leq (1+O(\varepsilon))r^*$.

For (b), we use [6, Lemma 2.2], which shows that the following property holds for all but at most d choices of j: the ball centered at q with radius $3r^*$ is contained in a quadtree box with side length at most $12kr^*$. By this property, $\mathcal{D}(p^*,q) \leq 12kr^*$, and so $r = \ell(p,q) \leq \ell(p^*,q) \leq d(p^*,q) + O(\varepsilon^2\mathcal{D}(p^*,q)) = (1+O(\varepsilon))r^*$. Then the ball centered at $c_{\varepsilon^2 r}(q)$ of radius $2r$ is contained in the ball centered at q of radius $(2+O(\varepsilon^2))r < 3r^*$, and is thus contained in $B_{12kr^*}(q)$.

(c) follows from (a) and (b), since $1 - d/k \geq 1 - O(\varepsilon)$ (and $d(S,q)$ does not depend on j). □

By (c), $\mathbb{E}_j[\mathbb{E}_S[\widetilde{\ell}(S,q)]]$ approximates $\mathbb{E}_S[d(S,q)]$ to within factor $1 \pm O(\varepsilon)$. It suffices to give an exact algorithm for computing $\mathbb{E}_S[\widetilde{\ell}(S,q)]$ for a query point q for a fixed j; we can then return the average, over all k choices of j.

4.2 The data structure: a BBD tree

We use a version of Arya et al.'s balanced box decomposition (BBD) tree [3]. We form a binary tree T of height $O(\log n)$, where each node stores a cell, the root's cell is the entire universe, a node's cell is equal to the disjoint union of the two children's cells, and each leaf's cell contains $\Theta(1)$ points of M. Every cell B is a difference of a quadtree box (the *outer box*) and a union of $O(1)$ quadtree boxes (the *holes*). Such a tree can be constructed by forming the compressed quadtree and repeatedly taking centroids, as described by Arya et al. (in the original BBD tree, each cell has at most 1 hole and may not be perfect hypercubes). We will store $O(1/\varepsilon^{O(1)})$ amount of extra information (various expectation and probability values) at each node. The total space is $O(n/\varepsilon^{O(1)})$.

4.3 An exact query algorithm for $\widetilde{\ell}$

In this section, we describe the algorithm for estimating $\mathbb{E}_S[\widetilde{\ell}(S,q)]$, given a query point q. First we extend the definition of $\widetilde{\ell}$ slightly: let $\widetilde{\ell}(S,q,r_0) = [\ell(S,q) \leq r_0] \cdot [\ell(S,q) \text{ is } q\text{-good}] \cdot \ell(S,q)$.

Consider a cell B of T and a query point $q \in B$. Let $R(B^c,q)$ denote the set of all possible values for $\ell(p,q)$ over points p in B^c, the complement of B. We solve the following extension of the query problem (all probabilities and expectations are with respect to the random subset S):

Problem 4.3. *For every* $r_0 \in R(B^c,q)$, *compute the values* $\Pr[\ell(S \cap B, q) > r_0]$ *and* $\mathbb{E}[\widetilde{\ell}(S \cap B, q, r_0)]$.

It suffices to compute these values for $r_0 \leq \sqrt{d}|B|$, where $|B|$ denotes the maximum side length of B, since they don't change as r_0 increases beyond $\sqrt{d}|B|$.

Lemma 4.4. *The number of elements in* $R(B^c,q)$ *that are below* $\sqrt{d}|B|$ *is* $O(1/\varepsilon^{2d})$.

Proof. If p is inside a hole H of B, then $\mathcal{D}(p,q) \geq |H|$, so we can consider a grid of side length $\Theta(\varepsilon^2|H|)$ and round p to one of the $O(1/\varepsilon^{2d})$ grid points without affecting the value of $\ell(p,q)$.

If p is outside the outer box of B, then $\mathcal{D}(p,q) \geq |B|$, so we can round p using a grid of side length $\Theta(\varepsilon^2|B|)$. In this case the number of grid points for $d(p,q) \leq \ell(p,q) \leq \sqrt{d}|B|$ is $O(1/\varepsilon^{2d})$ as well. □

We now describe the query algorithm. The base case when B is a leaf is trivial. Let B_1 and B_2 be the children cells of B. Without loss of generality, assume that $q \in B_2$ (i.e., $q \notin B_1$). We apply the following formulas, based on the fact that $\ell(S \cap B, q) = \min\{\ell(S \cap B_1, q), \ell(S \cap B_2, q)\}$ and that $S \cap B_1$ and $S \cap B_2$ are independent:

$$\Pr[\ell(S \cap B, q) > r_0] \;=\; \Pr[\ell(S \cap B_1, q) > r_0] \cdot \Pr[\ell(S \cap B_2, q) > r_0]; \tag{2}$$

$$\begin{aligned}
&\mathbb{E}[\widetilde{\ell}(S \cap B, q, r_0)] \\
&= \sum_{r \leq \sqrt{d}|B_2|} \Pr[\ell(S \cap B_1, q) = r] \cdot \mathbb{E}[\widetilde{\ell}(S \cap B_2, q, \min\{r, r_0\})] \;+ && (3)\\
&\quad \sum_{r \leq \sqrt{d}|B_2|} \Pr[\ell(S \cap B_1, q) = r] \cdot \Pr[\ell(S \cap B_2, q) > r] \cdot [r < r_0] \cdot [r \text{ is } q\text{-good}] \cdot r && (4)\\
&\quad + \Pr[\ell(S \cap B_1, q) > \sqrt{d}|B_2|] \cdot \mathbb{E}[\widetilde{\ell}(S \cap B_2, q, r_0)] && (5)\\
&\quad + \mathbb{E}\left[[\ell(S \cap B_1, q) > \sqrt{d}|B_2|] \cdot \widetilde{\ell}(S \cap B_1, q, r_0)\right] \cdot \Pr[S \cap B_2 = \emptyset]. && (6)
\end{aligned}$$

(3) and (5) cover the case when $\ell(S \cap B_2, q) \leq \ell(S \cap B_1, q)$, and (4) and (6) cover the case when $\ell(S \cap B_1, q) < \ell(S \cap B_2, q)$. For (5), note that $\ell(S \cap B_2, q) \leq r_0$ already implies $S \cap B_2 \neq \emptyset$ and $\ell(S \cap B_2, q) \leq \sqrt{d}|B_2|$.

By recursively querying B_2, we can compute all probability and expectation expressions concerning $S \cap B_2$ in (2)–(6). Note that $r_0 \in R(B^c, q) \subseteq R(B_2^c, q)$, and in the sums (3) and (4), it suffices to consider $r \in R(B_2^c, q)$ since $S \cap B_1 \subset B_2^c$. In particular, the number of terms with $r \leq \sqrt{d}|B_2|$ is $O(1/\varepsilon^{2d})$, as already explained. For the probability and expectation expressions concerning $S \cap B_1$, we examine two cases:

- Suppose that q is inside a hole H of B_1. For all $p \in B_1$, $\mathcal{D}(p,q) \geq |H|$ and $\ell(p,q) \geq \Omega(\varepsilon^2|H|)$, so we can consider a grid of side length $\Theta(\varepsilon^4|H|)$ and round q to one of the $O(1/\varepsilon^{4d})$ grid points without affecting the value of $\ell(p,q)$, nor affecting whether $\ell(p,q)$ is q-good. Thus, all expressions concerning $S \cap B_1$ remain unchanged after rounding q. We can precompute these $O(1/\varepsilon^{O(1)})$ values for all grid points q (in $O(n/\varepsilon^{O(1)})$ time) and store them in the tree T.

- Suppose that q is outside the outer box of B_1. For all $p \in B_1$, $\mathcal{D}(p,q) \geq |B_1|$, so we can consider a grid of side length $\Theta(\varepsilon^2|B_1|)$ and round each point $p \in M \cap B_1$ to one of the $O(1/\varepsilon^{2d})$ grid points without affecting the value of $\ell(p,q)$. Duplicate points can be condensed to a single point by combining their probabilities; we can precompute these $O(1/\varepsilon^{2d})$ probability values (in $O(n)$ time) and store them in the tree T. We can then evaluate all expressions concerning $S \cap B_1$ for any given q by brute force in $O(1/\varepsilon^{O(1)})$ time.

Since the height of T is $O(\log n)$, this recursive query algorithm runs in time $O((1/\varepsilon^{O(1)}) \log n)$. Therefore we arrive at the main result of this section.

Theorem 4.5. *Given a stochastic set of n points in a constant dimension d, we can build an $O(n/\varepsilon^{O(1)})$-space data structure in $O((1/\varepsilon^{O(1)})n \log n)$ time, so that for any query point, we can compute a $(1+\varepsilon)$-factor approximation to the expected nearest neighbor distance in $O((1/\varepsilon^{O(1)}) \log n)$ time.*

5 Conclusion

We show that even elementary proximity problems become hard under the stochastic model, and point to a need for new techniques to achieve approximation bounds. On one hand, the intractability may seem unsurprising because the computations involve an exponential number of possible subsets. But as we showed in a related work [10], for several fundamental geometric structures computing the expectation is possible in polynomial time despite the overt summation of exponentially many terms. However, it was also shown in the same paper that computing the expected length of the minimum spanning tree is $\#P$-Hard. Our current work continues the research begun in [10] and attempts to delineate between computationally tractable and intractable for the most basic of the geometric problems. We believe that our results can serve as building blocks for a theory of geometric computation under a stochastic model of input.

References

[1] P. Afshani, P. K. Agarwal, L. Arge, K. G. Larsen, and J. M. Phillips. (Approximate) Uncertain Skylines. In *ICDT*, pages 186–196, 2011.

[2] P. K. Agarwal, S.-W. Cheng, Y. Tao, and K. Yi. Indexing Uncertain Data. In *PODS*, pages 137–146, 2009.

[3] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions. *J. ACM*, 45:891–923, 1998.

[4] T. M. Chan. Approximate Nearest Neighbor Queries Revisited. *Discrete and Computational Geometry*, 20:359–373, 1998.

[5] T. M. Chan. Closest-point Problems Simplified on the RAM. In *Proc. SODA*, pages 472–473, 2002.

[6] T. M. Chan. Polynomial-time Approximation Schemes for Packing and Piercing Fat Objects. *J. Algorithms*, 46:178–189, 2003.

[7] M. De Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 2008.

[8] L. J. Guibas, J. S. B. Mitchell, and T. Roos. Voronoi Diagrams of Moving Points in the Plane. In *Proc. 17th International Workshop on Graph-Theoretic Concepts in Comptuter Science*, volume 570, pages 113–125. LNCS, 1992.

[9] G. H. Hardy, G. Polya, and J. E. Littlewood. *Inequalities*. Cambridge Press, 1952.

[10] P. Kamousi, T. M. Chan, and S. Suri. Stochastic Minimum Spanning Trees in Euclidean Spaces. In *Proceedings of the 27th annual ACM symposium on Computational geometry*, SoCG '11, pages 65–74.

[11] D. A. Klain and G. Rota. *Introduction to Geometric Probability.* Cambridge, 1997.

[12] D. E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching.* Addison-Wesley, 1973.

[13] M.-S. Lin and Y.-J. Chen. Counting the Number of Vertex Covers in a Trapezoid Graph. *Inf. Process. Lett.*, 109:1187–1192, 2009.

[14] M. Löffler and M. J. van Kreveld. Largest and Smallest Convex Hulls for Imprecise Points. *Algorithmica*, 56(2):235–269, 2010.

[15] J. S. Provan and M. O. Ball. The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. *SIAM J. Comput.*, 12(4):777–788, 1983.

[16] S. P. Vadhan. The Complexity of Counting in Sparse, Regular, and Planar Graphs. *SIAM Journal on Computing*, 31:398–427, 1997.

[17] L. Valiant. Universality Considerations in VLSI Circuits. *IEEE Trans. Computers*, 30:135–140, 1981.

[18] L. G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[19] M. J. van Kreveld, M. Löffler, and J. S. B. Mitchell. Preprocessing Imprecise Points and Splitting Triangulations. *SIAM J. Comput.*, 39(7):2990–3000, 2010.