SOME RESULTS IN COMPUTATIONAL AND COMBINATORIAL GEOMETRY

BY MUDASSIR SHABBIR

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

William Steiger

And approved by

New Brunswick, New Jersey

October, 2014

ABSTRACT OF THE DISSERTATION

Some Results in Computational and Combinatorial Geometry

by Mudassir Shabbir

Dissertation Director: William Steiger

In this thesis we present some new results in the field of discrete and computational geometry. The techniques and tools developed to achieve these results add to our understanding of important geometric objects like line arrangements, and geometric measures of depth.

Small Hitting Sets

Given a set S of n points, a *weak ϵ-net* X is a set of points (not necessarily in S) such that any convex set, called a range, that contains more than an ϵ fraction of S must meet X for a fixed $\epsilon > 0$ [30]. Aronov *et al.* gave the first bounds on ϵ when the cardinality of X is a fixed small number in the plane. Later Mustafa and Ray proved that $|X| = 2$ can be chosen so that we hit all convex ranges that contain $4n/7$ points of S [46]. We describe an $O(n \log^4 n)$ time algorithm to find points $z_1 \neq z_2$, at least one of which must meet any convex set of "size" greater than $4n/7$; z_1 and z_2 comprise a hitting set of size two for such convex ranges. This is the first algorithm for computing the hitting sets of fixed size.

Data Depth

Data-depth measures are real valued functions that are defined on the points of $\mathbb{R}^d$ with respect to a given set S in $\mathbb{R}^d$. They are helpful in nonparametric statistical analysis by partitioning the space in a center-outwardly fashion. We introduced a new framework to study many well-known data-depth measures in a uniform way. We define and provide first bounds for *line-depth* and show how it bridges the relation among Tukey-depth, simplicial-depth, and ray-shooting depth measures in $\mathbb{R}^3$. We also develop the first algorithm to efficiently compute a point of *high* ray-shooting depth in the plane.

Graph Search with Immunity

Faults and viruses often spread in the networked environments by propagating from a site to neighboring site. We model this process of *network contamination* by using graphs. Consider a graph $G = (V, E)$, whose vertex set is contaminated. Our goal is to decontaminate the set $V(G)$ using the mobile agents that move along the edge set of G. The *temporal immunity* $\tau(G) \geq 0$ is defined as the time that a decontaminated vertex of G can remain continuously exposed to a contaminated neighbor without getting infected itself. We study the lower and upper bounds on the temporal immunity required to decontaminate some classes of graphs - mostly geometric - that correspond to some well-known network topologies, and we present an upper bounds on $\iota_1(G)$, in some cases with matching lower bounds.

Acknowledgements

I am grateful to Boris Aronov, William Steiger, Jeff Kahn, and Mario Szegedy for serving on my thesis review committee. Their feedback improved the quality of this thesis manifold.

I would like to thank my advisor William Steiger for standing by my side through all this time. For his continuous support and mentoring, I would always remain grateful.

For my training as a researcher, I would like to acknowledge the crucial part played by my teachers. I must thank Jeff Kahn, Bill Steiger, Mario Szegedy, József Beck, Mike Saks, Vladimir Retakh, Muthu, and Greg Cherlin for their guidance and patience. I would like spell out my gratitude for Jeff Kahn for offering the series of courses on *Combinatorics* - one couldn't ask for a more exhilarating hour and twenty minutes.

I am thankful to Nabil Mustafa for introducing me to some interesting problems in the area and for tons of delightful conversations.

I would like to thank my friends Asif, Shakeel, Arzoo, Pavel, Rajat, Sergiu, Rezwana, Basit, Edinah, Zhiyuan, Imdad, Ben, Fatma, Amey, Charlene, Jess, Mashariq, Haroon, Talal, Sharjeel and Ahmed (and others who are slipping through my mind) for the inspiring discussions and nice times that we had together. I am indebted to all my friends for their contributions direct or indirect. I also thank the amazing folks who live on the third floor of Hill Center including Carole, Regina, Maryann, and Aneta for their kindness.

In the end, I acknowledge the support of my family, specially that of my parents, for this endeavor. I find it remarkable in a community where sending kids to school is thought of as a *lousy investment*!

Dedication

To "Jon Snow" who knows nothing!

Table of Contents

Chapter 1

Introduction

A famous theorem in combinatorics states that if we have "enough" points in general position (no three on a line) in the plane, then n of them must be in convex position. In fact let $f(n)$ denote the smallest value - if any - for which the previous statement is true, Erdős and Szekeres actually proved in 1935 that the function f exists, and that it is at most $1 + \binom{2n-4}{n-2}$, roughly $O(4^n/\sqrt{n})$. They later showed in 1961 that $f(n)$ is at least $1 + 2^{n-2}$, actually believed to be the correct value for $f(n)$, and highlighting the large gap that still exists between known upper and lower bounds for f.

This far-reaching fundamental fact about geometry and combinatorics has stimulated a great deal of subsequent research, and actually gave birth to Ramsey theory. It could also be said that this theorem exhibits the spirit of discrete geometry, and that it exemplifies its strong connection with Computational Geometry: if we were given a set of points in general position in the plane, *How* do we *find* the largest subset in convex position? The journal *Discrete and Computational Geometry*, the flagship for this part of Combinatorics and Computer Science, is devoted to research that expands both subjects and in doing so, highlights the strong interconnections.

The following four chapters of this thesis are - each in their own particular ways - examples of these kinds of close connections between Combinatorics and Computation. In Chapter 2 we give the first efficient algorithm to compute small hitting sets for convex ranges. In order to accomplish this, it was necessary to deepen our understanding of line arrangements in the plane, and thus, this chapter represents something *more* than the algorithm itself. It is hoped that the new insights into the structure of line arrangements will help develop efficient algorithms for other problems as well. Chapters 3 and 4 both depend on new algorithmic and combinatorial tools for arrangements of points and lines in $\mathbb{R}^2$. Finally, Chapter 5 contains several algorithms - and the combinatorics behind them - for certain graph traversal problems.

In Chapter 2 we show how to find small hitting sets for convex ranges. Given a set S of n points, a *weak ϵ-net* X is a set of points (not necessarily in S) such that any convex set, called a range, that contains more than an ϵ fraction of S must meet X for a fixed $\epsilon > 0$ [30]. Because every range must meet X, X is also called a "hitting set". It is well known that weak ϵ-nets of constant size always exist. Aronov *et al.* studied the problem of finding the best bounds on ϵ for fixed small cardinality X [4]. When X is just a single point the problem is a well known one. A classical result known as the centerpoint theorem states that for any set S of n points, there exists a point x that meets all convex ranges that contain more than $2n/3$ points of S [53].

Aronov *et al.* gave the first bounds on ϵ when X has two, three, four, and five points. Later Mustafa and Ray proved that $|X| = 2$ can be chosen so that we hit all convex ranges that contain $4n/7$ points of S [46]. They also gave an example to show that it was the best possible. Their proof relies on the existence of a "highest-lowest" point - defined in with respect to convex ranges that contain $4n/7$ points of S.

We describe an $O(n(\log n)^4)$ algorithm to find points $z_1 \neq z_2$, at least one of which must meet any convex set of "size" (for a convex set we abuse the term *size* to mean the size of the intersection of this set with S) greater than $4n/7$; z_1 and z_2 comprise a hitting set of size two for such convex ranges. And the fraction $4/7$ is minimal, as shown in [47]. This algorithm can then be used to construct (i) three points, one of which must meet any convex set of size $> 8n/15$; (ii) four points, one of which must meet any convex set of size $> 16n/31$; (iii) five points, one of which must meet any convex set of size $> 20n/41$. The efficient algorithm - the first for small hitting sets for convex ranges - is not the only contribution for Chapter 2. In order to devise the efficient algorithm and to demonstrate its complexity - a more detailed understanding of line arrangements was needed. Chapter 2 ends with a discussion of some open algorithmic and combinatorial problems suggested by these results.

Data-depth measures are real valued functions that are defined on the points of $\mathbb{R}^d$ with respect to a given set S in $\mathbb{R}^d$. The set S represents a discrete sample from an unknown distribution.

John Tukey was one of the first mathematicians to extend the notion of data-depth to $\mathbb{R}^d$, $d > 1$. He began with the observation that for n given points in $\mathbb{R}^1$, and x in $\mathbb{R}^1$, $d(x)$ is the number of $s_i \in S$ in the smaller of the two halflines containing x. He then defined for $z \in \mathbb{R}^d$ the halfspace depth, whereby $d(z) = \min_{z \in h}(|\{s_i \in h\}|)$, the min taken over all halfspaces of $\mathbb{R}^d$ that contain z, so $d(z) \leq (n + d)/2$, just as was the case for $d = 1$. Halfspace depth, or Tukey depth, is one of the most familiar and widely used depth notions.

Again given a set S of n data points in $\mathbb{R}^d$, the simplicial depth of $z \in \mathbb{R}^d$ is the number of subsets of $d + 1$ points of S that are in convex position and that contain z. A median is a point in $\mathbb{R}^d$ of maximal simplicial depth; a point z not within $Conv(S)$ has depth $d(z) = 0$.

Finally given a point set S in a d dimensional space, the *ray-shooting depth* of a point $q \in \mathbb{R}^d$ is the smallest number of the $(d-1)$-simplices intersected by any ray from q where a simplex is induced on any d points in S. The ray-shooting depth of the point set S is defined as the maximum ray-shooting depth of any point in $\mathbb{R}^d$. In the chapter 3, we introduce a new depth notion - the line depth. We study its relation to the other three data-depth measures defined above and prove the first nontrivial bounds on the depth of a median. Inspired by this connection, we propose a new framework to study the data-depth measure in a uniform manner. We think this may facilitate better understanding of the relationships between the different notions.

In Chapter 4 we present a new algorithm to compute a point of “high” ray-shooting depth. We describe the algorithm and derive its complexity. The ray-shooting Theorem says:

Theorem 1. *[26]. Any set S of n points in $\mathbb{R}^2$ has RS-depth at least $n^2/9$, and this bound is* *tight in the worst case.*

The topological proof given in [26] follows from a variant of Brouwer's fixed point theorem and is as such purely existential, although a straightforward algorithm could be derived from it with running time $O(n^5 \log^5 n)$ by an exhaustive search. We present instead an algorithm that computes a point $z \in \mathbb{R}^2$ of ray-shooting depth $n^2/9$. Its running time is $O((n^2 \log^2 n)$. We also discuss a software package that we created for the famous statistical computing environment R using the concept for ray-shooting depth.

Finally in Chapter 5, a certain graph search problem is discussed. We define this problem in the network terminology. Faults and viruses often spread in the networked environments by propagating from a site to neighboring site. We model this process of *network contamination* by using graphs. Consider a graph $G = (V, E)$, whose vertex set is contaminated and our goal is to decontaminate the set $V(G)$ using the mobile decontamination agents that move along the edge set of G. The *temporal immunity* $\tau(G) \geq 0$ is defined as the time that a decontaminated vertex of G can remain continuously exposed to a contaminated neighbor without getting infected itself. We study the lower and upper bounds on the temporal immunity required to decontaminate some classes of graphs - mostly geometric - that correspond to some well-known network topologies and present an upper bounds on $\iota_1(G)$, in some cases with matching lower bounds. Variations of this problem have been extensively studied in the literature. But the proposed algorithms have been restricted to *monotone* strategies, where a vertex, once decontaminated, may not be recontaminated. We exploit the *nonmonotonicity* of some new strategies to give the bounds which are strictly better than those derived using the monotone strategies.

Instead of a chapter to address prospects for further research, we have taken up these issues at the end of each separate chapter.

Chapter 2

Hitting Large Convex Ranges

2.1 Introduction

Let S be a set of n given points in general position in $\mathbb{R}^2$. If A is a convex subset of $\mathbb{R}^2$, its "size" is defined to be $|A \cap S|$, the number of points of S that it contains. The (Tukey) depth, $d(z)$, of a point $z \in \mathbb{R}^2$ is defined as the minimum, over all halfspaces h containing z, of $|S \cap h|$, the size of the smallest halfspace containing z. It is known that there always exists a point $z \in \mathbb{R}^2$ (not necessarily in S) with depth $d(z) \geq n/3$. Such a point is called a *centerpoint* for S. The constant $c = 1/3$ is best possible: for every $c > 1/3$ there are sets S with respect to which no point has depth cn. The interesting algorithm of Jadhav and Mukhopadhyay [31] computes a centerpoint in linear time.

Alternatively, if z is a centerpoint for S, *every* convex set of size greater than $2n/3$ *must* contain z. A centerpoint may thus be said to "hit" all convex subsets of $\mathbb{R}^2$ with more than $2/3$ of the points of S. For this reason, centerpoint z is called a *hitting-set (of size* 1*)* for convex sets of size greater than $2n/3$. Mustafa and Ray [46], following related work of Aronov *et al.* [4], studied the possibilities for hitting sets with more than one point, a natural extension of the notion of centerpoint. They showed that given $S \subseteq \mathbb{R}^2$ there are points $z_1 \neq z_2$ (not necessarily in S) such that every convex set of size greater than $4n/7$ must meet at least one of them. In addition they showed via a construction that the constant $4/7$ is best possible for hitting sets of size 2: for every $c < 4/7$ there are sets S for which, whatever points $x, y \in \mathbb{R}^2$ be chosen, there is a convex subset containing more than cn points of S, but containing *neither* x nor y).

Earlier, Aronov *et al.* [4] had shown that the optimal constant c was in the interval $[5/9, 5/8]$.

Let $c_k \in (0,1)$ be the smallest constant for which, for every set S of n points in $\mathbb{R}^2$, there are distinct points $z_1, \ldots, z_k$, at least one of which must meet any convex set of size greater than $c_k n$. We know $c_1 = 2/3$ and $c_2 = 4/7$. Mustafa and Ray were also able to show that $c_3 \in (5/11, 8/15]$, that $c_4 \leq 16/31$ and that $c_5 \leq 20/41$.

Here we address the leading algorithmic question: given S, how can we *find* a small hitting set for it, and what is the complexity of this task? Some answers are contained in the following statements, and in the proofs.

Theorem 1. *Let S be a set of n given points in general position in $\mathbb{R}^2$ and take $c_2 = 4/7$. Then in $O(n(\log n)^4)$, we can find distinct points z_1, z_2, at least one of which must meet any convex set of size greater than $c_2 n$.*

The running time is in the unit cost RAM model.

As in Mustafa and Ray, and using Theorem 1 inductively, we also show

Corollary 1. *As in Theorem 1, in $O(n(\log n)^4)$ we can find points*

1. z_1,z_2,z_3, *one of which must meet any convex set of size greater than* $8n/15$*;*
2. z_1,z_2,z_3,z_4, *one of which must meet any convex set of size greater than* $16n/31$*;*
3. z_1,z_2,z_3,z_4,z_5, *one of which must meet any convex set of size greater than* $20n/41$.

In what follows, we describe the algorithms to support these statements, and conclude with some open questions that are suggested by our results, and seem especially interesting.

2.2 The Hitting-Set Algorithms

Again, S denotes a set of n given points in $\mathbb{R}^2$, and if A is a subset of $\mathbb{R}^2$, its "size" is defined as $|A \cap S|$, the number of points of S that it contains. Consider the collection $\mathcal{R}$ of all convex subsets of size greater than $c_2 n$, with $c_2 = 4/7$. These are our *ranges*. To prove Theorem 1, we describe an $O(n(\log n)^4)$ algorithm to construct a pair of points z_1, z_2, at least one of which must meet every range in $\mathcal{R}$.

For every pair $A \neq B$ in $\mathcal{R}$ consider $A \cap B$ and note that $|A \cap B| > n/7$. Write $p_{A,B} = (u, v) \in A \cap B$ for a point of minimal y-coordinate; if $A \cap B$ is not bounded below, $v = -\infty$. The existence proof in [46] showed that z_1 may be taken as a point $p_{A',B'} = (u, v)$ such that $p_{A',B'}$ has the maximum y-coordinate over all pairs $A \neq B$ in $\mathcal{R}$ i.e., a point in the intersection of two ranges whose lowest-point is the highest. Such a point is called a *highest lowest-point*. Then the second point z_2 may then be taken as a (usual) centerpoint for $S \setminus (A' \cap B')$ and that everything works out as claimed, that is every range meets either z_1 or z_2 (or both).

Let $p = (u, v)$ be a lowest-point in $A' \cap B'$, the intersection of two ranges, each of size more than $c_2 n$, and where v is as large as possible. Our proof of Theorem 1 partly relies on understanding what such a point dualizes to in the line arrangement dual to S. This is combined with tools introduced by Matoušek [42] for computing the center of a given point set, along with some combinatorial and algorithmic observations. All together, they enable us to show that z_1 can be found in the stated complexity. Once we have z_1, z_2, a centerpoint for $S \setminus (A' \cap B')$, can then be found using the well-known algorithm of Jadhav and Mukhopadyay in linear time [31].

2.2.1 Characterizing a Highest Lowest-Point

Given the set S of n points in general position in the plane, we will work with $\mathcal{L}$, the n lines dual to S using $v = xu + y$ as the equation of line that is dual to point (x, y), and $(-m, b)$ as the point dual to the line with equation $y = mx + b$. In the *arrangement* $\mathcal{A}(\mathcal{L})$ of the n dual lines, the j^{th} level λ_j is the closure of the set of points in $\mathbb{R}^2$ that lie on a single line of $\mathcal{L}$ and have exactly j lines of $\mathcal{L}$ *on* or *above* them, $1 \leq j \leq n$. Γ_j denotes the boundary of the convex hull of λ_j. It is clearly a concave function if $j > n/2 + 1$ and a convex function if $j < n/2 - 1$.

Notation 1. *From here on we will reserve k to express the critical size of ranges that allow* *two-point hitting sets for every set of n points in general position in $\mathbb{R}^2$, that is,*

$$k := 1 + \lfloor 4n/7 \rfloor. \tag{2.1}$$

Thus, if A and B are convex sets, each of which contain k points of S, $A \cap B$ is a convex set with at least $n/7$ points of S. Also we observe that

Fact 1. *We can narrow the focus to convex sets A, B which are actually closed halfspaces of size k, each supported by some point in S.*

That's because for every pair A, B of ranges whose intersection is bounded below, $A' \equiv \text{conv}(A \cap S) \subseteq A$ and $B' \equiv \text{conv}(B \cap S) \subseteq B$, so the pair A', B' must have a lowest-point at least as high as that of A, B. In addition, a lowest-point of $A' \cap B'$ is supported by a line through an edge of A' and a line through an edge of B' and if *either* of these two halfspaces has size $> k$, one or both of the supporting lines can be moved in such a way that both halfspaces will now have size k and their intersection will have a higher lowest point . It's also clear that if either of the supporting lines does not meet some point of S, the line may be shifted up or down until it *does* meet a point of S and so that the lowest point would be higher.

From now on, we will only consider halfspaces of size k for our ranges.

Suppose then that $p_{A,B} = (u_{AB}, v_{AB})$ is a lowest point in $A \cap B$. There are two distinct cases: either

- **Case 1 (above/above):** Both of the ranges are halfspaces of size k, each supported below by a line containing at least one point of S. One of the lines (say ℓ_A) has slope $m_A \geq 0$ and the other has slope $m_B \leq 0$), and there is at most one zero slope .

- **Case 2 (above/below:)** Again, both ranges are halfspaces of size k, each supported by a line containing at least one point of S. One of the ranges (say B) is supported above by a line with slope $m_B > 0$ and A is supported below by a line with slope m_A, and $m_B > m_A \geq 0$. A second variant of this situation is when $m_B < m_A \leq 0$ (not shown).

We first characterize the dual of a highest lowest point when the ranges A, B are in the above/above case. Write $y = m_A x + b_A$ for the equation of ℓ_A, the line of support for A, $y = m_B x + b_B$ for the equation of ℓ_B, the line of support for B, and suppose without loss of generality that $m_A > 0$ and $m_B \leq 0$. Since A and B both have k points of S on or above, and since $p_{A,B}$ is a highest lowest point, it is necessary that (i), there are k points of S on or above both ℓ_A and ℓ_B, and each contains a point of S, (ii) $p_{A,B}$ is the intersection of ℓ_A and ℓ_B, and (iii) $p_{A,B} = (u, v)$ has the largest value of v among any such pair A, B of ranges for which (i) and (ii) both hold. For (iii) to hold it is also clear that each line in fact contains two points of S. If not, the line may be rotated about its *one* point so as to give a higher intersection point $p_{A,B}$. These observations reveal that

1. ℓ_A dualizes to a point $q_A = (u_A, v_A)$ that is a vertex of λ_k, the k-level of $\mathcal{A}(\mathcal{L})$, *and* with $u_A > 0$, and ℓ_B dualizes to $q_B = (u_B, v_B)$ also a vertex of λ_k, but with $u_B \leq 0$,
2. $p_{A,B}$ dualizes to a line through two points having the properties described above, and
3. the above line meets the vertical axis $u = 0$ at a point $(0, v)$ with the maximal possible value of v.

In view of the fact that the convex hull of the k-level of $\mathcal{A}(\mathcal{L})$ is a concave function, the three properties above prove

Lemma 1. *The dual of a highest lowest point for the above/above case is the line ℓ incident with the edge of Γ_k that crosses the vertical axis $u = 0$. So if ℓ has equation $v = xu + y$ the point $p = (x, y)$ is a highest lowest point for the set of above/above ranges.*

In [42] Matoušek gave an algorithm to compute the tangent to a given level (for us it's k) in an arrangement of n lines that crosses a given vertical line (for us it's $u = 0$) and it runs in $O(n(\log n)^3)$. Combined with Lemma 1, it follows that

Lemma 2. *If a highest lowest point $p_{A,B}$ arises from a pair of ranges that are above/above, it can be found in $O(n(\log n)^3)$.*

The above/below situation is more complex, and much harder to deal with. The dual of a lowest point here is, as before, straightforward to describe, but now it is more challenging to characterize a highest one in such a way that it may be found efficiently.

There are again two sub cases: (i) in the first, one range , say A, is *above* its line of support ℓ_A that has slope $m_A \geq 0$, while range B is *below* its line of support ℓ_B, and its slope is $m_B > m_A$; in (ii), the other sub-case, again A is above ℓ_A and B below ℓ_B, but now $m_B < m_A < 0$. As before, if $\ell_A \cap \ell_B = p_{A,B}$ is to be a highest lowest point, both lines must be incident with two points of S.

In the first sub-case, ℓ_A dualizes to a vertex $q_A = (u_A, v_A)$ on λ_k, the k^{th} level of $\mathcal{A}(\mathcal{L})$, and $u_A \geq 0$. The dual of ℓ_B is a point $q_B = (u_B, v_B)$ with $u_B > u_A$ and q_B has k lines of $\mathcal{L}$ on or *below* it, so it is a vertex of λ_{n-k+1}, the $n-k+1$ level in $\mathcal{A}(\mathcal{L})$ and it lies "to the right" of q_A. Finally, $p_{A,B}$ dualizes to a line $\ell_{A,B}$ incident with two such points, and which meets the vertical axis at a point $(0, v)$ with the maximal possible value for v.

The other case where both slopes are less than 0 is symmetric. Here the dual of a highest lowest point is a line $\ell_{A',B'}$ joining a vertex $q_{B'} = (u_{B'}, v_{B'})$ on λ_{n-k+1}, to a vertex $q_{A'} = (u_{A'}, v_{A'})$ on λ_k, $u_{B'} < u_{A'} < 0$, and where such a line meets the vertical axis at a point $(0, v)$ with the largest possible value of v. To summarize:

Lemma 3. *The dual of a highest lowest-point in the above/below case is either (i) a line $\ell_{A',B'}$ joining a vertex $q_{B'} = (u_{B'}, v_{B'})$ on λ_{n-k+1}, to a vertex $q_{A'} = (u_{A'}, v_{A'})$ on λ_k, and where $u_{B'} < u_{A'} < 0$, or (ii) a line $\ell_{A,B}$ joining vertex $q_B = (u_B, v_B)$ on λ_{n-k+1}, to vertex $q_A = (u_A, v_A)$ on λ_k, and where $u_B > u_A \geq 0$. Among all such lines, if any, it is the one which meets the vertical axis $u = 0$ at the highest possible point.*

The remainder of the chapter addresses the algorithmic issues; we exploit the characterizations of highest lowest points in an efficient way. Lemma 2 already resolves the above/above case in a simple and straightforward manner.

But Lemma 3 suggests a search among *pairs* of points where in each pair, we have a vertex

$p = (u, v) \in \lambda_k$ and a vertex $p' = (u', v') \in \lambda_{n-k+1}$ - both in the positive halfspace or both in the negative halfspace, and p' further from the vertical axis $u = 0$. There could be more than $[n * 2^{c*\sqrt{\log n}}]^2$ pairs to check, by virtue of the lower bound construction for planar halving sets found by Géza Tóth [54]; on the other hand we know that there no more than $O(n^{8/3})$ such pairs, in view of Tamal Dey's bound on k-sets in $\mathbb{R}^2$ [22].

In what follows we will show that in fact there are *only a constant number* of such pairs that need to be checked. This is a main combinatorial and algorithmic contribution of this chapter. It allows us to use adaptations of Matoušek's algorithms to construct convex hulls of levels in arrangements with which we will discover and evaluate only a constant number of the relevant pairs and thus find a highest lowest-point in the claimed running time.

2.2.2 Finding the Best Above/Below Candidate - Separated Case

The centerpoint theorem states that there exists a point z such that for any line ℓ through z there are at least $n/3$ points of a given point set S on both sides of ℓ. This implies that in the dual Γ_j and Γ_{n-j+1} can always be separated by a line, as long as $j \leq n/3$. In our situation it may or may not be the case that Γ_k and Γ_{n-k+1} intersect. But if these convex functions do not meet, or if they meet in at most one point, there are only few possibilities for a highest lowest-point, and they may be checked efficiently. In particular,

Lemma 4. *If* Γ_k *and* Γ_{n-k+1} *meet in at most one point, a highest lowest-point may be found in* $O(n(\log n)^4)$

Proof. Using Lemma 1, in $O(n(\log n)^3)$ we compute the line ℓ incident with that edge of Γ_k that crosses $u = 0$. This is the dual of the above/above candidate for highest lowest-point. Next we use Matoušek's $O(n(\log n)^4)$ algorithm [42] to actually construct Γ_k and Γ_{n-k+1} and in a further $O(n \log n)$, we can learn whether the convex sets (above Γ_{n-k+1} and below Γ_k) can be separated by a line. If *yes*, then the above/above candidate ℓ and the two inner tangents to Γ_k

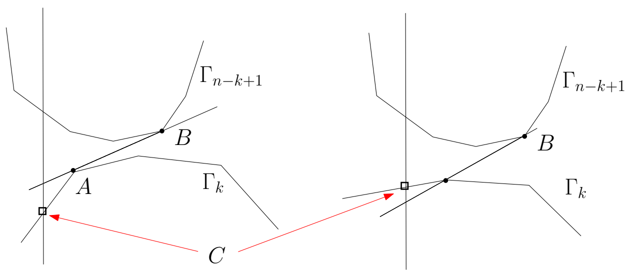

Figure 2.1: Above/Below where the hulls of level k and level $n-k+1$ are disjoint, or touch at one point.

and Γ_{n-k+1} suffice to describe the set of possible candidates for the dual of a highest lowest-point. The reason, as suggested in Fig. 2.1, is that a lowest-point B on λ_{n-k+1} and a highest point A on λ_k, (with A and B both on the same side of the vertical axis $u = 0$ but with B further away), must both be vertices of the convex hulls of their levels if the line they determine is to meet $u = 0$ at the highest possible point (see Fig. 2.1, left-side). The other possibility is that the above/above candidate gives a higher lowest-point, and this occurs if the inner tangent has its two vertices on opposite sides of the vertical axis (Fig. 2.1, right-side). □

We will refer to the process of checking for this configuration, and then finding its candidate for highest lowest-point, as **THE PHASE I ALGORITHM.**

2.2.3 Finding the Best Above/Below Candidate - Crossing Case

Finally we come to the real crux of the small hitting set computation problem. This is the complicated case where Γ_k and Γ_{n-k+1} actually meet in *two* points. First we define some notation.

Notation 2. *A vertical strip between a pair of points* $A = (A_x, A_y)$ *and* $B = (B_x, B_y)$ *is denoted by* $|A, B|$. *Formally*

$$|A, B| = \{p = (p_x, p_y) \in \mathbb{R}^2 : A_x \le p_x \le B_x\}.$$

Also,

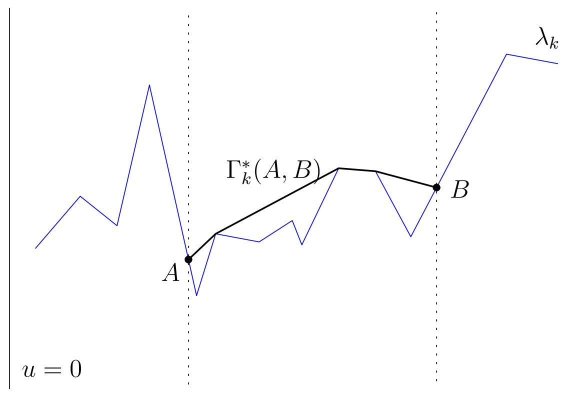

Figure 2.2: The area between two dotted vertical lines represents the vertical strip $|A, B|$. The fat black curve is $\Gamma^*_k(A, B)$, the boundary of the restricted convex hull.

Definition 1. *Given a vertical strip $|A, B|$ with $A = (A_x, A_y)$ and $B = (B_x, B_y)$, an important idea is the convex hull of λ_k , restricted to $|A, B|$, which we write as $\Gamma^*_k(A, B)$ or just Γ^*_k when A and B are clear from the context. It is the upper chain of the convex hull of the set of vertices of λ_k that lie interior to $|A, B|$ along with $\lambda_k(A_x)$ (the point on λ_k with horizontal coordinate A_x) and $\lambda_k(B_x)$. Similarly $\Gamma^*_{n-k+1}(A, B)$ is the restricted hull of λ_{n-k+1} for $|A, B|$, the lower chain of the hull of the vertices of λ_{n-k+1} strictly within the interior along with the points $\lambda_{n-k+1}(A_x)$ and $\lambda_{n-k+1}(B_x)$.*

It is not difficult to adapt Matoušek's original $O(n(\log n)^4)$ algorithms for this context but we postpone the details until section 2.3.

Notation 3. *We write $\mu(A, B)$ for the vertical coordinate of the point at which the line through a pair of points A, B meets the vertical line $u = 0$.*

We begin with an interesting and possibly unexpected structural feature that motivates and supports our algorithm and the analysis of its running time. It is expressed in the following statement.

Fact 2. *There are at most 7 edges $\overline{A_i A_{i+1}}$ of the convex hull Γ_k which are touched or crossed by λ_{n-k+1}. Given Γ_k, they may be found in $O(n \log n)$.*

Proof of Fact 2: Let $E = \overline{A_iA_{i+1}}$ be an edge of Γ_k that is touched or crossed by λ_{n-k+1}, $A_i = (u_i, v_i)$ and $A_{i+1} = (u_{i+1}, v_{i+1})$. We will say λ_{n-k+1} "pierces" Γ_k in this edge.

The piece-wise linear curve λ_{n-k+1} is above E at u_i and also at u_{i+1}, so if it pierces E there is a point $A' = (u', v') \in E$ $(u' > u_i)$ where λ_{n-k+1} first meets E. Since A_i is in level k and A' is in level $n - k + 1$, we know that between A_i and A', at least $n/7$ more lines of $\mathcal{L}$ crossed edge E from above than from below, and the same holds for the segment on E from A' to A_{i+1}, except "above" and "below" are reversed. This implies that E meets at least $2n/7$ lines from $\mathcal{L}$. The claim now follows by virtue of the fact that Γ_k is convex, so no line of $\mathcal{L}$ can meet it more than twice, and this means there can be at most 7 such edges.

For each line $\ell \in \mathcal{L}$ it can be decided whether it meets Γ_k in $0, 1$, or 2 of its edges - and which ones - in $O(\log n)$ using binary search on the vertices of Γ_k. Also, by this same process, we know for each of the edges in Γ_k whether it might have enough lines meeting it to allow λ_{n-k+1} to pierce it. In a further $O(n \log n)$ we can actually check the at most seven candidate edges by "walking" the edge from left to right and keeping track of the levels of $\mathcal{A}(\mathcal{L})$ that are crossed. □

Remark 1. Though we won't use it, we observe that by the same reasoning used to prove Fact 2, we can also show that at most 7 edges of Γ_{n-k+1} can be below a vertex of λ_k in the vertical strip defined by that edge, and they too may be found in $O(n \log n)$. Furthermore Fact 2 is also true for the restricted convex hulls $\Gamma^*_k(A, B)$ and $\Gamma^*_{n-k+1}(A, B)$ for any fixed pair of points A and B.

We begin by assuming we have done the $O(n(\log n)^4)$ time processing to obtain Γ_k, and Γ_{n-k+1}, and have discovered the configuration in Figure 2.4, with L and R the left and the right intersection points of Γ_k and Γ_{n-k+1} respectively. Refer to a pair of points $C = (u_1, v_1) \in \lambda_k$ and $D = (u_2, v_2) \in \lambda_{n-k+1}$ $0 \leq u_1 < u_2$ as a *legal pair*. Likewise an *illegal pair* is a pair of the points, one in the λ_k and the other in the λ_{n-k+1}, that is not legal. We seek a legal pair of the point C, D with the property that $\mu(C, D)$ is the highest among all such pairs.

Let C and D be a given pair of points as above with the additional condition that at least one of the points lies outside the vertical strip $|L, R|$. Then we claim that the algorithm in PHASE I can be easily adapted to compute the best legal pair of points C^* and D^* in further $O(n(\log n)^4)$.

CASE A: When either C or D lies outside the Strip $|L, R|$. There are four possible cases depending on whether C (or D) is to the left of $|L, R|$ or to the right. We observe that in all of these four possible cases we can meet the prerequisite for the algorithm in PHASE I that the convex hulls of the levels do not meet at more than one point.

When D *lies to the right of* R, we take the convex hull of the points of λ_k that lie to the right of R, denoted by $\Gamma_k^*(R, R')$ where $R' = (\infty, 0)$. The restricted convex hull $\Gamma_k^*(R, R')$ does not meet Γ_{n-k+1} except possibly at the point R as shown in part (i) of Figure 2.3. Similarly when D *lies to the left of* L take the convex hull of of points in λ_k that are to the left of L, $\Gamma_k^*(L, L')$, where $L' = (0, 0)$. Two convex hulls $\Gamma_k^*(L, L')$ and Γ_{n-k+1} meet at at most one point L as in Figure 2.3(ii).

The other two cases when C lies outside the strip $|L, R|$ instead of D can be symmetrically handled in the straightforward way, using same ideas. Therefore in four runs of the algorithm in PHASE I the best pair among the four cases when either C or D lies outside the Strip $|L, R|$ can be computed.

So now assume we know that the optimal pair of points C and D lie in the vertical strip $|L, R|$. As above $L = (a, b)$ and $R = (c, d)$ denote the leftmost and rightmost intersection points of Γ_k and Γ_{n-k+1}. Without loss of generality we will assume that both $0 < a < c$ lie in the positive halfspace, and we will discuss only this case. If $a < 0$, or if $c < 0$, the discussion is easily adapted from the present context in an obvious way and we will omit the straightforward details. By Fact 2, at most seven edges of Γ_k are pierced by λ_{n-k+1} within the vertical strip $|L, R|$.

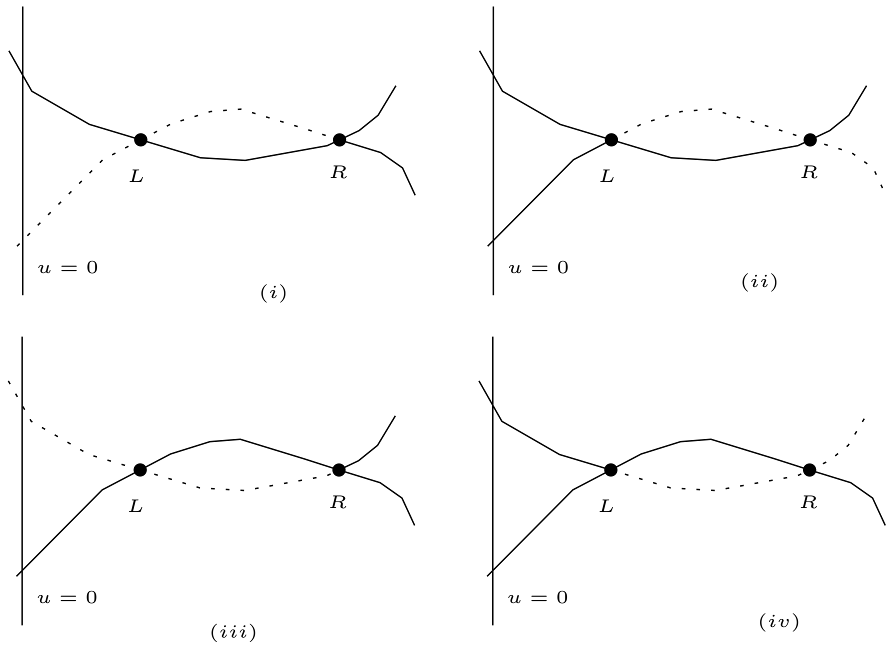

Figure 2.3: There are four cases: in (i) assume D lies to the right of R so we ignore all the points on Γ_{n-k+1} to the left of R, i.e., the dotted curve and convex hulls of the rest meet at only R. Similarly in (ii), (iii), and (iv).

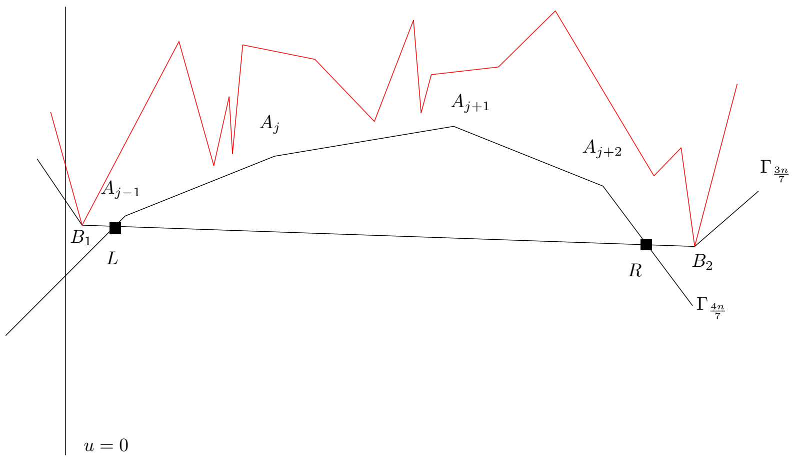

Figure 2.4: Γ_k and Γ_{n-k+1} meet in two points.

In $O(n \log n)$ we find the at most seven edges of Γ_k between L and R that are pierced by λ_{n-k+1}. We will refer to these edges as *bad edges* and recursively search the vertical strips defined by these edges for the optimal pair of points. An edge that is not bad is a *good edge*. We argue that that the vertical strips defined by the endpoints of a good edge can't contain one or both of the optimal pair of points and hence can be ignored except for one special case.

CASE B: A Corner Case. As already discussed, we know that the total number of bad edges on the convex hull Γ_k is at most seven. On the other hand, if all the edges are good, we consider it as a special case. The configuration in Figure 2.4 is an illustration of this case. We handle this case, the "corner case", as follows.

First of all, observe that this can only happen when there are no vertices of λ_{n-k+1} below the boundary of Γ_k and only a single edge of Γ_{n-k+1} is intersected by Γ_k. Let $\overline{B_1 B_2}$ be that edge on Γ_{n-k+1}. Rotate the vertical line through B_2 (the right endpoint of this edge) counter-clockwise until it hits Γ_k at some vertex, say, A_i as illustrated in Figure 2.5(i). Note that the one of the optimal pair of points, C, must lie on Γ_k but the other optimal point D that is to the right of C may lie anywhere on λ_{n-k+1} and not necessarily on the convex hull Γ_{n-k+1}. Furthermore any point on Γ_k that is to the left of A_i can't be in an optimal pair because the line through any such point $A_j, j < i$ and a point $B \in \lambda_{n-k+1}$ which is to the right of A_j would hit the vertical axis lower than $\mu(A_i, B_2)$, the point where the line through A_i, B_2 meets the vertical axis which is a potential optimal pair. As before let the vertices of Γ_k be ordered from left to right and let A_ℓ be the last vertex on Γ_k above the line through B_1, B_2. We observe that if there is a legal pair of points C, D in this vertical strip with $\mu(C, D) \geq \mu(A_i, B_2)$ then it follows that (1) $\mu(C, D) \leq \mu(A_{\ell-1}, A_\ell)$ and (2) $\mu(C, D) \geq \mu(A_i, A_{i+1})$. We use this observation to perform a binary search for the optimal line.

Consider the line through $A_{\lfloor \frac{i+\ell}{2} \rfloor}, A_{\lfloor \frac{i+\ell}{2} \rfloor + 1}$. If λ_{n-k+1} intersects this line to the right of $A_{\lfloor \frac{i+\ell}{2} \rfloor}$ and to the left of B_2 then there is a legal pair of points C, D and a line through this pair has a y-intercept that is at the least as high as $\mu(A_{\lfloor \frac{i+\ell}{2} \rfloor}, A_{\lfloor \frac{i+\ell}{2} \rfloor + 1})$. This situation is

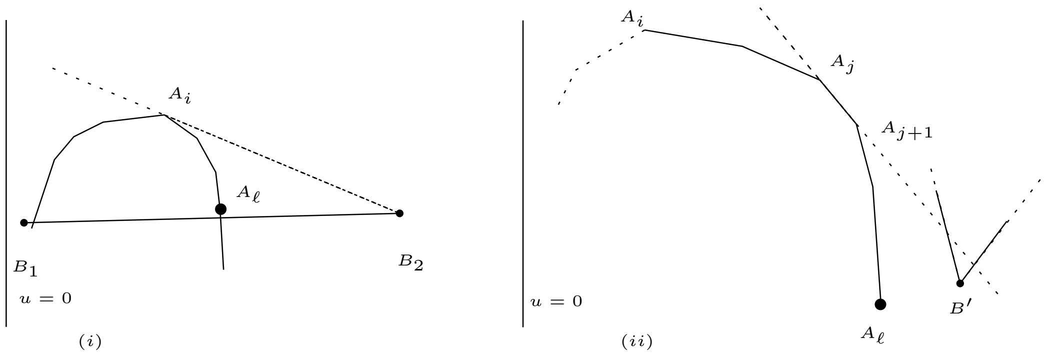

Figure 2.5: In (i) $\overline{B_1B_2}$ is an edge of Γ_{n-k+1} that is pierced by Γ_k. The line through A_i, B_2 is a tangent line to the convex hull Γ_k. In (ii) There is a vertex $B' \in \lambda_{n-k+1}$ with $\mu(B', A_{j+1}) \geq \mu(A_j, A_{j+1})$ if and only if the line through A_j, A_{j+1} is met by λ_{n-k+1} to the right of A_{j+1}. The test can be performed in $O(n \log n)$ time. A_ℓ is the rightmost vertex on Γ_k above $\overline{B_1B_2}$.

illustrated in Fig. 2.5(ii). We will recursively search for this pair in the strip $|A_{\lfloor \frac{i+\ell}{2} \rfloor + 1}, A_\ell|$. Otherwise we know that there is no such pair and the line through the optimal pair C, D has smaller y-intercept and we should look for it among the legal pairs of points in the vertical strip $|A_i, A_{\lfloor \frac{i+\ell}{2} \rfloor}|$. This test can be performed in $O(n \log n)$ time using Lemma 6. And in at the most $\lceil \log n \rceil$ tests, we can find the vertex D on in Γ_k^* that is in the best pair. To find the other point C in the optimal pair we draw a tangent from D to the λ_{n-k+1} to the right. This can be accomplished in $O(n \log^2 n)$ time using Lemma 7. Thus the algorithm runs in $O(n \log^2 n)$ time in this case and finds the best pair in the vertical strip $|B_1, B_2|$.

Fact 3. *Vertical strips of good edges need not be checked as long as there is a bad edge to the right.*

Proof. Let $\overline{A_iA_{i+1}}$ be an arbitrary good edge on Γ_k such that there is a bad edge $\overline{A_jA_{j+1}}$ to the right. The vertices $A_i, A_{i+1}, \ldots$ on Γ_k are ordered from the left to right and $i + 1 \leq j$. Let B be a point on the level λ_{n-k+1} in the vertical strip $|A_j, A_{j+1}|$, and since $\overline{A_jA_{j+1}}$ is a bad edge, we can assume that B is below the edge $\overline{A_jA_{j+1}}$. By the convexity of Γ_k, B also lies below the line through A_i, A_{i+1}. Therefore line through B, A_{i+1} meets $u = 0$ at a higher point than the line through A_i, A_{i+1}.

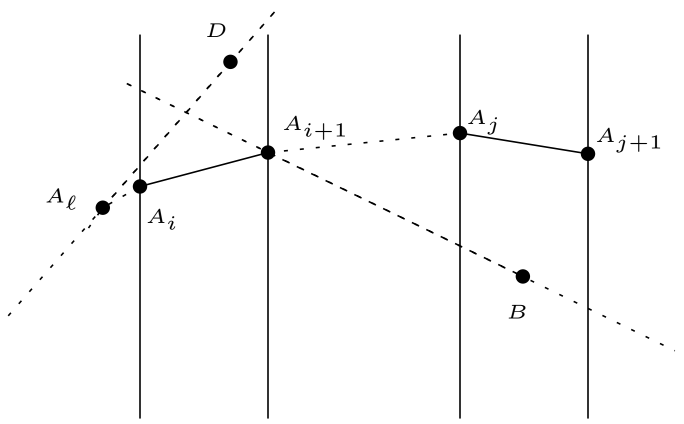

Figure 2.6: $\overline{A_iA_{i+1}}$ is a good edge and there is a bad edge $\overline{A_jA_{j+1}}$ to the right. The points B and D are vertices of λ_{n-k+1}. Since $\mu(B, A_{i+1}) \geq \mu(D, A_\ell)$ for all A_ℓ on Γ_k that lie to the left of D, the vertical strip $|A_i, A_{i+1}|$ of the good edge may be skipped when searching for a pair of points with the highest μ value.

By the assumption that $\overline{A_iA_{i+1}}$ is a good edge, any point $D \in \lambda_{n-k+1}$ in the vertical strip $|A_i, A_{i+1}|$ must lie above the line through A_i, A_{i+1}. This implies that $\mu(D, A_\ell) \leq \mu(A_i, A_{i+1})$ for all A_ℓ on Γ_k that are to the left of D. By transitivity it follows that $\mu(B, A_{i+1}) \geq \mu(D, A_\ell)$ for any point D on λ_{n-k+1} in the vertical strip $|A_i, A_{i+1}|$. The vertical strip defined by A_i, A_{i+1} need not be considered while searching for a pair with a line of highest y-intercept. □

Remark 2. The statements in Lemma 4, Fact 2, Fact 3 and the arguments in the CASE A and CASE B all remain true when we replace the convex hulls Γ_k and Γ_{n-k+1} with the restricted convex hulls $\Gamma^*_k(U, V)$ and $\Gamma^*_{n-k+1}(U, V)$ for an arbitrary pair of points U and V.

We will invoke the following recursive algorithm with the initial input $U = (0, 0)$ and $V = (\infty, 0)$.

PAIR(U,V)

- The **INPUT** is a pair of points $U = (a, a')$ and $V = (b, b')$, $a < b$. The initial call is with $U = (0, 0)$ and $V = (\infty, 0)$.

- The **OUTPUT** is two vertices $C = (u_1, v_1) \in \lambda_k$ and $D = (u_2, v_2) \in \lambda_{n-k+1}$, both in the vertical strip erected through U and V, and with $0 \leq u_1 < u_2$. They have the property that among all such pairs within the strip, the line through C and D meets $u = 0$ in the highest possible point.

- The **STEPS:**

  1. Construct $\Gamma^*_k(U, V)$, the restricted convex hull of λ_k, and $\Gamma^*_{n-k+1}(U, V)$.
  2. If $\Gamma^*_k(U, V)$ and $\Gamma^*_{n-k+1}(U, V)$ meet at at most one point then apply PHASE I and return the best pair. Otherwise continue.
  3. Compute L and R, the left and the right intersection points of two convex hulls.
  4. Assuming that at least one of the points C and D lies outside vertical strip, solve the four possible cases using PHASE I and compute the best pair among them as described in CASE A.
  5. Construct $\Gamma^*_k(L, R)$ - the restricted convex hull of λ_k - and $\Gamma^*_{n-k+1}(L, R)$.
  6. Find all the bad segments $\overline{A_i A_{i+1}}$ on $\Gamma^*_k(L, R)$.
  7. If there are no bad segments then we are in the corner case; handle it as described in CASE B and find the optimal pair in this case and jump to the last STEP.
  8. Let $\overline{A_j A_{j+1}}$ be the rightmost bad edge. If there are any more edges to the right of A_{j+1} then recursively call PAIR with A_{j+1} and V' as input where V' is the point where λ_{n-k+1} intersects the vertical line through V. Note that this would yield a case in which the convex hulls don't intersect or if they do intersect then it's just a corner case since there can't be any bad segments in this strip.
  9. For each bad segment $\overline{A_i A_{i+1}}$:
     - Compute the restricted hull $\Gamma^*_{n-k+1}(A_i, A_{i+1})$.
     - Rotate the vertical line through A_i counter-clockwise until it first becomes tangent to λ_{n-k+1} at a point $X = (s, t), s \in [A_i, A_{i+1}]$ within the strip (this

may be done in $O(n)$ time by finding the smallest slope among lines from U to the vertices of $\Gamma^*_{n-k+1}(A_i, i+1)$).

  – Recursively solve the left and the right subproblem: PAIR (A_i, X) and PAIR (X, A_{i+1}).

10. Return the best pair among all cases.

END PAIR.

Lemma 5. *The algorithm PAIR(U,V) correctly finds a pair* $C = (u_1, v_1) \in \lambda_k$ *and* $D = (u_2, v_2) \in \lambda_{n-k+1}$*, both in the vertical strip erected through* U *and* V*, and with* $0 \leq u_1 < u_2$*. They have the property that among all such pairs within the strip, the line through* C *and* D *meets* $u = 0$ *in the highest possible point.*

Proof of Lemma 5: We have already discussed the correctness of our algorithm when the convex hulls of two levels don't meet at more than one point or when the convex hulls of two levels do intersect but there are no bad edges. The following two claims complete the proof of the lemma. □

Fact 4. *The optimal pair can not lie in vertical strips defined by two different bad edges.*

Proof. Assume $\overline{A_iA_{i+1}}$ and $\overline{A_jA_{j+1}}$ are a pair of bad edges on Γ^*_k where $i < j$, vertices being ordered from the left to right. For all the points in λ_{n-k+1} that lie in the vertical strip $|A_i, A_{i+1}|$, the points in λ_k that lie in $|A_j, A_{j+1}|$ are illegal pairs. So we only need to consider the case for pairs with a point $B \in \lambda_{n-k+1} \cap |A_j, A_{j+1}|$ and the other point $A \in \lambda_k \cap |A_i, A_{i+1}|$. But for any fixed A and B we note that $\mu(B, A_j) \geq \mu(A, B)$. This completes the proof. □

Fact 5. *While splitting a bad edge* $\overline{UV}$ *into two subproblems, the optimal pair lies inside the left subproblem or the right subproblem.*

Proof. By virtue of the choice of the point $X = (s,t) \in \Gamma^*_{n-k+1}$ that splits $|U, V|$ into $|U, X|$ and $|X, V|$ (it's the lower point of tangency from $U \in \lambda_k$ to Γ^*_{n-k+1}) either the pair $C_L =$

$(x, y) \in \lambda_k$, $D_L = (x', y') \in \lambda_{n-k+1}$, returned by the left sub-problem or the pair $C_R = (z, w) \in \lambda_k$, $D_R = (z', w') \in \lambda_{n-k+1}$, $s < z < z$ returned in the right subproblem must determine a line meeting $u = 0$ at a point higher than that by any pair, one point on λ_{n-k+1} to the right of X and one point on λ_k to the left of X. Indeed, this underlies the ability to ignore the evaluation at all but a constant number of the lines through pairs of points, one on λ_k the other on λ_{n-k+1}, while still discovering the dual of a highest lowest-point. □

Finally we complete the proof of Theorem 1 as we establish

Fact 6. *Let L and R denote the two points where Γ_k and Γ_{n-k+1} meet. The algorithm* **PAIR(L,R)** *finds the dual of a highest lowest-point in the above/below case in $O(n(\log n)^4)$.*

Proof of Fact 6: The key idea behind the proof is to provide a concrete integer bound (56) on the depth of the recursion. We establish this bound by following a longest path to a leaf, a node that we will call z. As we follow the path from the root to z, at each level, j, we will define a segment e_j for that node in such a way that after we reach z the collection of segments defined along this path will be vertically separated. More important, we will establish the following properties:

1. at least $n/7$ lines of the arrangement must meet each segment,

   BUT

2. no line in $\mathbb{R}^2$ can meet more than eight of the segments in our collection.

We will combine these two properties to obtain the bound on the depth of the recursion.

For the details, we imagine running PAIR on a single edge $\overline{A_iA_{i+1}}$ of Γ_k that is pierced by λ_{n-k+1}. As mentioned, z denotes the leaf node on a longest path in the recursion tree of the algorithm on our n lines. This lowest node corresponds to a subproblem on a vertical strip between $U = (a, b)$ and $V = (c, d)$. These points are in $\Gamma_k^*(U, V)$ and λ_{n-k+1} does *not* penetrate it, so PHASE I will find the pair for a highest lowest-point. We fix a point $q \in \lambda_{n-k+1}$ strictly within the open strip $|U, V|$.

Now, we trace the path from the root of the recursion tree down to node z. At the root (level 1), the problem is divided by the edges on the convex hull Γ_k. Good edges and corner cases are resolved and each bad edge is divided into two subproblems. Since there are at most seven bad edges, there are at most fourteen children of the root node. If the point q lies inside a vertical strip defined by either a good edge or a corner case, the optimal pair is searched for as described in the discussion on CASE A and CASE B and no further recursive calls are made. So without loss of generality, q lies in the vertical strip of a bad edge $\overline{A_i A_{i+1}}$ of Γ_k. And X_1, the point in λ_{n-k+1} that splits this into the two sub-problems, is the point of lower tangency from A_i to λ_{n-k+1} that lies within the strip. If the path from this node to z follows the *right* subproblem $[X_1, A_{i+1}]$, we define segment e_1 to be $\overline{A_i X_1}$ as illustrated in Fig. 2.7(i) and Fig. 2.7(ii). Otherwise we take $e_1 = \overline{Z_1 A_{i+1}}$, where $Z_1 \in \lambda_{n-k+1}$ is the point of *lower* tangency from A_{i+1} to λ_{n-k+1} that lies within the strip as in Fig. 2.7(iii) and Fig. 2.7(iv).

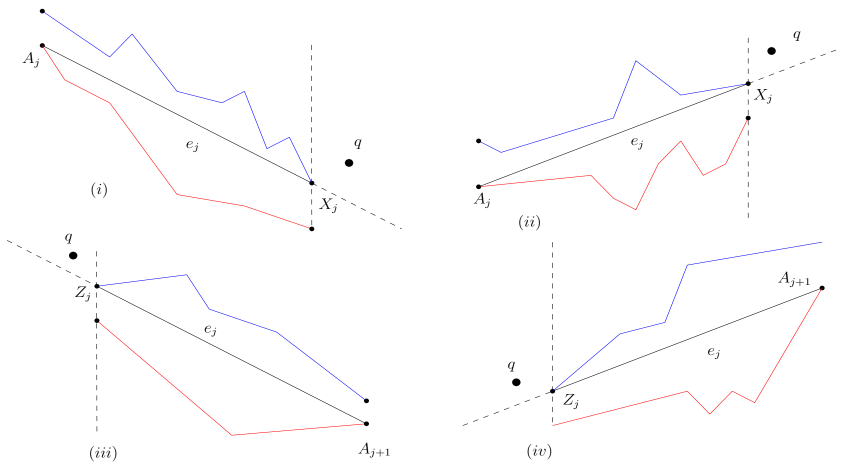

Figure 2.7: The segment e_j lies on a line that is tangent from a point on Γ_k (either A_j or A_{j+1}) to λ_{n-k+1} to the right or to the left. The four possibilities are illustrated in the figure.

In level m of the recursion we are between vertices A_j and A_{j+1} of Γ_k^* and edge $\overline{A_j A_{j+1}}$ is penetrated by λ_{n-k+1}. If X_j is the point in λ_{n-k+1} of lower tangency to λ_{n-k+1} from A_j that lies within this strip, and if the path to z now follows the right subproblem $[X_j, A_{j+1}]$, then we define the segment e_j to be $\overline{A_jX_j}$. Otherwise the path to z follows the left subproblem and e_j will be the segment $\overline{Z_jA_{j+1}}$, where $Z_j \in \lambda_{n-k+1}$ is the point of lower tangency to but now, from A_{j+1}. Fig. 2.7 illustrates the edge for level j of the recursion; it matches a vertex in λ_k to one in λ_{n-k+1}. Both vertices are in the opposite subproblem in this node from the path leading to leaf z, so this edge e_j is vertically separated from all subsequent edges $e_i, i > j$, and this shows that *all* edges in our collection are vertically separated.

The final step now will show that the depth of the recursion is at most 56 - there just cannot exist more segments that satisfy all their properties (one endpoint on λ_k, the other at a point of tangency to λ_{n-k+1}, as well as being vertically separated from all subsequent edges). First, if some line ℓ meets edge e_j in our collection ℓ may meet (i) *both* the k-level and the $n-k+1$-level, (ii) *neither* of these levels, or (iii) exactly one of these levels, within e_j's vertical strip. In the first case, $n/7$ lines of $\mathcal{L}$ cross the given line ℓ within this strip, and because the edges are vertically separated, there can be at most seven such edges. If ℓ crosses e_j but misses *one* or *both* λ_k and λ_{n-k+1} within this strip, it's in a different branch of the recursion from the deepest node z, so ℓ misses all subsequent edges $e_i, i > j$.

There are four possible cases for the line segment e_j as illustrated in Fig 2.7. In the first case on the top left of the figure, the line through edge e_j is a tangent from a point A_j on Γ_k to a point X_j on λ_{n-k+1}. This implies that all the points in the level λ_{n-k+1} lie above this line and so does its (current *and* any future) restricted convex hull. In particular intersection of two convex hulls (restricted to the strip between X_j and A_{j+1}) lies above this line. Since all the subsequent edges e_i in our collection with $i > j$ lie in this intersection, all of them lie above the line through e_j. Also by construction all the edges in our collection are vertically separated. Any line that intersects e_j but does not intersect the level λ_{n-k+1} in the vertical strip of e_j, the blue curve in the figure, must intersect the vertical line through X_j at a point below X_j and hence must not intersect any e_i with $i > j$. Argument is similar for the rest of the three cases.

So the worst case is that ℓ previously met seven edges and crossed both λ_k and λ_{n-k+1} with each of them, and now crosses e_j and misses all subsequent edges, and this gives 8 as the maximum number of edges in our collection that can meet any line. Since each edge in our collection meets at least $n/7$ lines of the arrangement, and none of these $n/7$ lines could meet more than eight edges, the collection has at most 56 edges. □

Remark 3. We note that this bound on the depth of the recursion implies that we do the above/below case in at most $c \cdot (14)^{56} n(\log n)^4$ time steps because at each step we split the problem into at most 14 subproblems.

2.3 Computing the Convex Hull of a Level Restricted to a Strip

In [42] Matoušek describes an algorithm for computing the convex hull of a level λ_k in an arrangement of n lines. It runs in $O(n \log^4 n)$ time. In this section, we provide a few simple modifications to that algorithm and give the proof of its correctness to make it work in the restricted settings, that is to compute the convex hull of a level λ_k restricted to a vertical strip. Modifications we recommend are quite simple in nature and are provided here solely for the purpose of completeness. We do assume familiarity with Matoušek's algorithm in [42] on reader's part.

Since a given set S of n points, the n lines dual to S can be computed in $O(n)$ time we will assume that the input to our algorithm is a pair of points U, V along with a collection $\mathcal{L}$ of n lines in the plane. Desired output is the convex hull of λ_k restricted to the strip $|U, V|$. The leftmost (similarly the rightmost) point on λ_k in this strip can be easily computed by considering the order in which the lines in $\mathcal{L}$ intersect the vertical line through U (similarly V). Therefore we assume without loss generality that U and V lie on λ_k. Modifications required in first three Lemmas are as below. As in Matoušek we assume that $k < n/2$ as the algorithm is symmetric for $k \geq n/2$.

Lemma 6. *Given a collection $\mathcal{L}$ of n lines, a line q, and two points U, V, one can find all vertices of λ_k lying on q restricted to vertical strip between U, V in time $O(n \log n)$.*

Proof. Compute all vertices of λ_k lying on q using Lemma 3.1 in [42] for the corresponding problem. Throw away the vertices lying to the left of U and the vertices lying to the right of V and return the rest of the vertices. Time complexity is dominated by Matoušek's $O(n \log n)$ procedure in Lemma 3.1. $\square$

Lemma 7. *Given a collection $\mathcal{L}$ of n lines and three points x, U, V, one can find tangent to $\Gamma_k^*(U, V)$ between U, V thru x and touching $\Gamma_k^*(U, V)$ to the right of x (if it exists) in time $O(n \log^2 n)$.*

Proof. Let τ^* be the tangent line that we seek. Although we don't know τ^* yet for a pair of lines ℓ, ℓ' we can decide the order in which they intersect τ^* by checking whether their intersection point y lies on/above or below τ^*; following the notation in [42] this decision problem is referred to as *the question* (ℓ, ℓ'). The question (ℓ, ℓ') can be decided by computing the points of λ_k that lie on the line through x and y in $O(n \log n)$ time using Lemma 6. It is easy to see that once we know the order of intersections of all the lines in $\mathcal{L}$ with τ^*, we will explicitly know the line τ^*.

Start by computing the order in which the lines in $\mathcal{L}$ intersect the vertical line through U and the order in which they intersect the vertical lines through V. If these orders are identical for a pair of lines, i.e., they do not intersect in the vertical strip $|U, V|$, then the order of their intersection with τ^* is of no consequence so we will ignore them. As in [42], the problem of sorting the order of the intersection of lines with τ^* is solved using Megiddo's parametric search [44].

Given a batch of questions $(\ell_1, \ell_1') \dots (\ell_m, \ell_m')$ where $m \le n/2$, at least half of these questions can be answered in $O(n \log n)$ time as follows. Slopes z_i of the lines through x, y_i are computed where y_i is the intersection point of (ℓ_i, ℓ_i') for all $1 \le i \le m$. Check whether z,

the median of z_i's, lies above or below τ^* in $O(n \log n)$ time using Lemma 6. If z lies above τ^* then all $z_j \geq z$ lie above as well and if it lies below the line τ^* then all $z_k \leq z$ lie below as well. In any case we can answer at least $m/2$ of the questions in $O(n \log n)$ time.
Cole's extension of the parametric search [18] implies that the search problem for τ^* can be solved in $O(\log n)$ weighted batched questions. We adapt the scheme above for the weighted batched problem exactly as in [42]. The overall time complexity for deciding the order of all lines and hence computing the tangent τ^* is $O(n \log^2 n)$ time.

□

As remarked in [42], in a special case of Lemma 7 when x is at infinity one can compute a tangent to a restricted λ_k that has a prescribed slope.

Lemma 8. *Given a collection $\mathcal{L}$ of n lines, a pair of points U, V, and a vertical line W lying inside $|U, V|$ one can compute tangent τ^* touching Γ_k^* at its intersection point with W, in time $O(n \log^3 n)$.*

Proof. There are two steps for the algorithm to find τ^* in this case. First, we can compute the slope of τ^* relative to the slopes of lines in $\mathcal{L}$ as below:

- Use Lemma 7 to find a tangent τ' to Γ_k^* parallel to a line q in $\mathcal{L}$.
- If τ' touches Γ_k^* strictly to the left of W we can push the intersection of τ' with W down by slightly lowering the right end of τ' and hence the slope of τ^* is smaller than the slope of q. We proceed similarly for τ' touching Γ_k^* to the right of W.

After the lines in $\mathcal{L}$ have been sorted by their slopes, a binary search can be performed to find the slope of τ^* relative to the slopes of lines in $\mathcal{L}$ in $O(n \log^3 n)$ time.

In the second step we explicitly find τ^* using a parametric search. Sort the intersection of the lines in $\mathcal{L}$ with τ^* - deciding the order of a pair of lines ℓ, ℓ' is referred to as the question (ℓ, ℓ'). Since τ^* passes through a pair of points in the vertical strip, we will only solve questions

(ℓ, ℓ') for which intersection point $x = \ell \cap \ell'$ lies inside $|u, v|$. We already know relative slope of τ^* with respect to the lines in $\mathcal{L}$ therefore we can decide the question (ℓ, ℓ') once we know whether the intersection $x = \ell \cap \ell'$ lies above or below τ^*. And that is readily available using Lemma 7. Similarly the weighted questions batch problem is solved as in Matoušek. $\square$

The bounds in Lemma 4.1, Lemma 4.2, and Lemma 4.3 and the main algorithm in Matoušek are same for the restricted convex hull problem and the time complexity analysis of Matoušek is valid for the restricted convex hull algorithm as well.

2.4 Final Remarks

This algorithm leaves open the challenge of finding small hitting sets that are *optimal* for a given set S. Let $c_k(S)$ denote the smallest constant for which there exist k distinct points in $\mathbb{R}^2$, at least one of which must meet any convex set of size $> c_k(S)|S|$. As already mentioned, Mustafa and Ray showed that $c_2(S) \leq c_2 = 4/7$ and that $4/7$ is minimal. Another interesting combinatorial question is to learn the exact values for c_3, c_4, c_5 and it would be interesting to know how many points are needed to hit every convex set containing half the points of a given set S. Also the algorithmic problem of determining $c_2(S)$ for a given S, and of finding (efficiently) two points that meet all ranges of size $c_2(S)n$ seems interesting and nontrivial.

Chapter 3

k-Centerpoints: A Generalization

3.1 Introduction

Given a set of n real numbers $S = \{s_1, s_2, \ldots, s_n\}$ the *rank* $\gamma(x)$ is the cardinality of the set $\{s_i : s_i < x, 1 \leq i \leq n\}$. If the elements in S are distinct (a reasonable assumption for almost all the real data), the rank function partitions the real line into $n+1$ disjoint subinterval. Define the *depth* for any $x \in \mathbb{R}$ as

$$d(x) \equiv min(|\{j : s_j \leq x\}|, |\{j : s_j \geq x\}|)$$

This is the minimum number of data points that must be touched when moving monotonically from x to $\pm\infty$. The range of the depth function is between 0 and $\lceil \frac{n}{2} \rceil$. The min and the max elements of S have depth 1 (if they are unique) and a median has the maximal depth among all $x \in \mathbb{R}$.

When the data are from an unknown probability distribution F, data-depth provides valuable information about F. Statisticians and other researchers who work directly with the data realized the utility of depth-by-rank in $\mathbb{R}^1$ and sought analogous for higher dimensions. John Tukey was one of the pioneers and proposed the generalization for $\mathbb{R}^d$ called *halfspace depth*, now also known as the *Tukey depth*. The idea of Tukey depth is based on the generalization of the fact that in $\mathbb{R}^1$, the depth $d(x)$ is the minimum size (the number of elements of S) of a halfspace that contains x.

In the past 50+ years, mathematicians, statisticians, computer scientists, and a variety of researchers who work with the massive data have all contributed to the subject of data-depth.

In this chapter we will describe and relate some of these results. Several basic questions will be addressed including different generalizations of the depth in $\mathbb{R}^d$, and the relation between them. The main contribution of this chapter is the definition of a new depth measure in $\mathbb{R}^3$, the *line depth*, and proving the first nontrivial bounds for a deepest point with respect the line depth. We discuss some ways in which line depth relates to other well known depths in the three dimensions. We then propose a general framework in which they ought to be studied more uniformly. We present a simple set of conjectures on the bounds of the depth of the deepest point in a uniform data-depth model that generalizes several previous results in this regard. Finally we (i) argue for why these might be true, (ii) prove these conjectures are true for $\mathbb{R}^2$, and (iii) discuss the first non-trivial bounds in dimensions more than 2, and, (iv) via more detailed arguments, derive better bounds in $\mathbb{R}^3$.

The subsequent chapter will be devoted to the algorithmic aspects of the data-depth problem.

3.2 Data-Depth Measures

data-depth measures are real-valued functions that are defined on the points of $\mathbb{R}^d$ with respect to a set S of n given data points in $\mathbb{R}^d$. The set S represents a discrete sample from an unknown distribution. We review below some well-known data-depth measures and summarize some of their interesting properties.

3.2.1 Tukey Depth

John Tukey was one of the first mathematicians to extend the notion of data-depth to $\mathbb{R}^d$, $d > 1$. He began with the observation that for n given points in $\mathbb{R}^1$, $d(x)$ is the number of $s_i \in S$ in the smaller of the two halflines containing x. He then defined for $z \in R^d$ the halfspace depth, whereby $d(z) = min_{z \in h}(|\{s_i \in h\})|$, the min taken over all halfspaces of $\mathbb{R}^d$ that contain z, so $d(z) \leq (n + d)/2$, just as was the case for $d = 1$.

A fundamental result in combinatorial geometry, a consequence of Helly's theorem, is the Centerpoint Theorem. It states that there exists a centerpoint namely a point $z \in \mathbb{R}^d : d(z) \geq \lceil n/(d+1) \rceil$ and that the term $\lceil n/(d+1) \rceil$ is the best possible in the worst case. In fact all such points form a convex body in $\mathbb{R}^d$ called "the center". The points on the boundary of the convex hull of S have depth 1. A *median* is a point $z \in \mathbb{R}^d$ of maximal depth. Its depth $d(z)$ is the depth of S.

3.2.2 Simplicial Depth

Let S be n given data points in $\mathbb{R}^1$. Define the simplicial depth of $x \in \mathbb{R}$ to be the number of 1-simplices with vertices in S that contain x; that is, the number of *intervals* with endpoints in S. Thus points $x < \min(S)$ and $y > \max(S)$ have depth 0 and a maximal depth point - a median of S - has depth at least $(n/2)^2$. A key observation is that simplicial depth and Tukey depth order points in the same way: $d(x) < d(y)$ in Tukey depth $\Leftrightarrow$ the same holds with respect to simplicial depth.

In dimension $d \geq 2$, given a set S of n data points, the simplicial depth of $z \in \mathbb{R}^d$ is the number of subsets of $d+1$ points of S that are in convex position and that contain z. A *median* is a point in $\mathbb{R}^d$ of maximal depth and a point z not within $Conv(S)$ has depth $d(z) = 0$. If z is a median, its depth is the depth of S.

The First Selection Lemma (see [43], pg. 207) in two dimensions states that given any set S in $\mathbb{R}^2$, there exists a point in plane that has the simplicial depth at least $n^3/27$. Regina Liu began the study of the statistical properties of this depth measure in [39]. Not surprisingly, there is a close relation between Tukey and simplicial depths. Wagner pointed out the following fact:

Lemma 1. *[55] Any point q with Tukey depth τn with respect to an n-point set in $\mathbb{R}^d$ has simplicial depth at least $\frac{(d+1)\tau^d - 2d\tau^{d+1}}{(d+1)!} \cdot n^{d+1} - O(n^d)$.*

This is an interesting relation because along with the Centerpoint Theorem, it already gives the bounds on c_d that are very close to the ones in first selection lemma in two dimensions. Wagner used this to establish improved bounds for the First Selection Lemma in higher dimensions. Unfortunately it was proved soon after that this approach of taking an arbitrary centerpoint and bounding its simplicial depth can't give the optimal bounds for the First Selection Lemma even when $d = 2$. In fact the suboptimal bounds in the above lemma are the best possible using this technique as shown below.

Theorem 1. *[14] There exists an n-point set S and a point q, both in $\mathbb{R}^d$, such that the Tukey depth of q is $\tau \cdot n$ and the simplicial depth is at most $\frac{(d+1)\tau^d - 2d\tau^{d+1}}{(d+1)!} \cdot n^{d+1} + O(n^d)$.*

We observe the following simple relation in two dimensions between the two depth measures:

Claim 1. *Given a set S of n points in $\mathbb{R}^2$ with Tukey depth τn, with $1/3 \leq \tau \leq 1/2$, the simplicial depth of S is at least $(-2\tau^3 + 2.5\tau^2 - \tau + 1/6) \cdot n^3$. Moreover this relation is optimal.*

In particular, the above claim together with the Centerpoint Theorem implies the First Selection Lemma in $\mathbb{R}^2$. We will use the following key Lemma.

Lemma 2 (Boros-Füredi [11]). *Given a set S of n points in $\mathbb{R}^d$, where Tukey depth of S is τn, there exists a point p with depth τn, and a set $\mathcal{H}$ of $d+1$ halfspaces $\{h_1, \ldots, h_{d+1}\}$, such that (i) $|h_i \cap S| = \tau n$, (ii) p lies on the boundary hyperplane of each h_i, and (iii) $h_1 \cup \ldots \cup h_{d+1}$ cover the entire $\mathbb{R}^d$.*

Lemma 2 in the plane gives a point q of depth $\tau \cdot n$ with three halfplane each containing exactly $\tau \cdot n$ points and that cover the whole plane between them. Let $S_i, 1 \leq i \leq 6$ be the six regions such formed as outlined in figure 3.1.

Let s_i be the number of points of S in the region S_i. It's easy to see that there are at least the following number of triangles that contain q:

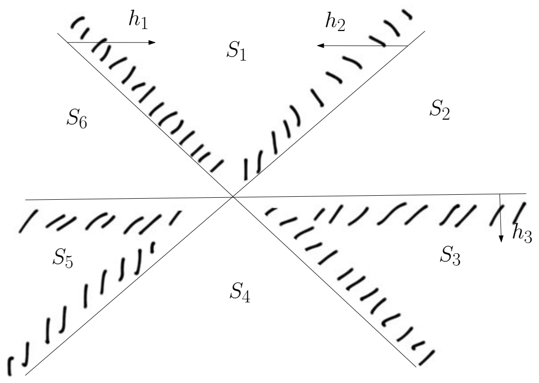

Figure 3.1: Three halfplanes h_1, h_2 and h_3 cover the whole plane. Regions are defined as follows: $S_1 = h_1 \cap h_2 \cap \overline{h}_3$, $S_2 = h_1 \cap \overline{h}_2 \cap \overline{h}_3$, $S_3 = h_1 \cap h_3 \cap \overline{h}_2$, $S_4 = h_3 \cap \overline{h}_1 \cap \overline{h}_2$, $S_5 = h_2 \cap h_3 \cap \overline{h}_1$, and $S_6 = h_2 \cap \overline{h}_1 \cap \overline{h}_3$.

- $s_5 \times s_2 \times (s_3 + s_4)$: with one point in S_5, other in S_2 and the third point in either $S_3 \cup S_4$ or $S_6 \cup S_1$,
- $s_1 \times s_4 \times (s_2 + s_3)$: with one point in S_1, other in S_4 and the third point in either $S_2 \cup S_3$ or $S_5 \cup S_6$,
- $s_3 \times s_6 \times (s_1 + s_2)$: with one point in S_3, other in S_6 and the third point in either $S_1 \cup S_2$ or $S_4 \cup S_5$,
- $s_1^3/3.0 + s_1^2 \times (1 - 2\tau)/2.0 - o(n^2)$: with all three points in $S_1 \cup S_4$,
- $s_3^3/3.0 + s_3^2 \times (1 - 2\tau)/2.0 - o(n^2)$: with all three points in $S_3 \cup S_6$,
- $s_5^3/3.0 + s_5^2 \times (1 - 2\tau)/2.0 - o(n^2)$: with all three points in $S_5 \cup S_2$, and
- $s_1 \times s_3 \times s_5 + s_2 \times s_4 \times s_6$: with one from each of S_1, S_3, and S_5 or one point from each of S_2, S_4, and S_6.

The claim follows by some algebraic manipulation. Basit *et al.* [7] used a more elaborate version of this technique to get improved bounds for the First Selection Lemma in $\mathbb{R}^3$.

While the First Selection Lemma gives the optimal bound on the simplicial depth of a point set in the plane, known bounds on the simplicial depth of point sets in higher dimensions are far from the optimal, despite the fact that the search for such bounds has been been subject of many major works in Discrete Geometry for last three decades. Bárány first showed that given a point set S of n points in $\mathbb{R}^d$ there exists a point q which is contained in $c_d.\binom{n}{d+1} - O(n^d)$ simplices with vertices from S where $c_d \geq \frac{1}{(d+1)^d}$ [5]. The bound on c_d was improved by Wagner to $\frac{d^2+1}{(d+1)^{(d+1)}}$ [55]. Recently Gromov showed that $c_d \geq \frac{2d}{(d+1)!(d+1)}$ using a new topological method [29].

When $d = 3$, Basit *et al.* proved that there is always a point in $\mathbb{R}^3$ which is contained in more than $0.00227n^4$ tetrahedrons induced by a set of n points. Gromov improved this bound to 0.07480 in [29]. In the same paper Gromov also gave a proof of the optimal bound of the First Selection Lemma using a topological method. This proof is of special interest because it establishes an optimal bound on *ray-shooting depth* in $\mathbb{R}^2$ that will be discussed in the next section.

3.2.3 Ray-Shooting Depth

Given a point set S in the plane, the *ray-shooting depth* of a point $q \in \mathbb{R}^2$ is the smallest number of the edges intersected by any ray from q where an edge is induced by a pair of points in S. The ray-shooting depth (henceforth called RS-depth) of S is the maximum RS-depth in $\mathbb{R}^2$. The ray-shooting Theorem is:

Theorem 2. *[26]. Any set S of n points in $\mathbb{R}^2$ has RS-depth at least $n^2/9$, and this bound is tight.*

It is observed that *any* point with RS-depth at least $n^2/9$ is both a centerpoint, as well as a point of high simplicial depth. Now let q be a point with RS-depth rn^2 with respect to a point set S in the plane. The Tukey depth of q is the minimum number of points of S contained in a halfplane that also includes q. Consider any line ℓ through q. Since q has RS-depth rn^2, each of the two half-infinite rays of ℓ starting at q must intersect rn^2 edges each. The line ℓ intersects at least $2rn^2$ edges. Say there are $k \leq n/2$ points of S on one side of ℓ and $n - k$ points on the other side. As $k \cdot (n-k) \geq 2rn^2$, we have $k \geq (n - n\sqrt{1-8r})/2$. Therefore both halfplanes defined by ℓ must contain at least $(n - n\sqrt{1-8r})/2$ points. The ray-shooting theorem states that $r \geq 1/9$ which shows that the Tukey depth of q is at least $n/3$ as required in the centerpoint theorem. Therefore the ray-shooting theorem implies the Centerpoint Theorem. As before let q be a point with RS-depth rn^2 with respect to a point set S in the plane . For the First Selection Lemma, we would like to count the number of triangles that contain q. For a point $s_i \in S$ consider the ray from q in the direction $\overrightarrow{q(-s_i)}$ ($-s_i$ is the antipodal point of s_i with respect to q). For every edge $\{s_j, s_k\}$ that intersects this ray, the triangle defined by $\{s_i, s_j, s_k\}$ must contain q where i, j, k are distinct. By the assumption on the ray-shooting depth of q, for each s_i there are at least rn^2 such triangles containing q. Summing up these triangles over all points s_i, where each triangle is counted three times, q lies in at least $rn^3/3$ distinct triangles.

But by the ray-shooting theorem, there exists a point with RS-depth $n^2/9$. Using $r = 1/9$ now gives the First Selection Lemma. So in dimension 2 the ray-shooting theorem implies the First Selection Lemma.

On the other hand, a centerpoint is not always a point of "high" ray-shooting depth. Consider the example in the figure 3.2. The centerpoint c in the figure has ray-shooting depth at most $n^2/18 + 6n$. It can be shown that this is the worst possible example - that a centerpoint always has ray-shooting depth of $n^2/18 - O(n)$.

Fact 1. *Given a set of points S in the plane any centerpoint c has ray-shooting depth more than $n^2/18 - O(n)$.*

Proof. It is enough to show that there are "enough" edges meeting an arbitrary ray ρ from the centerpoint c. Since a point in S contributes at most $n-1$ edges, we may assume that ρ meets a point s_1. Let the points in $S = \{s_1, s_2, \ldots, s_n\}$ be ordered radially around c in the counter-clockwise order. Translate and rotate the point set so that c is the origin and s_1 lies on the positive horizontal axis. Let ℓ_i be the line through c and the point s_i - so ℓ_1 is the horizontal axis with at least $n/3$ points above and at least $n/3$ points below. Note that for a point s_i with

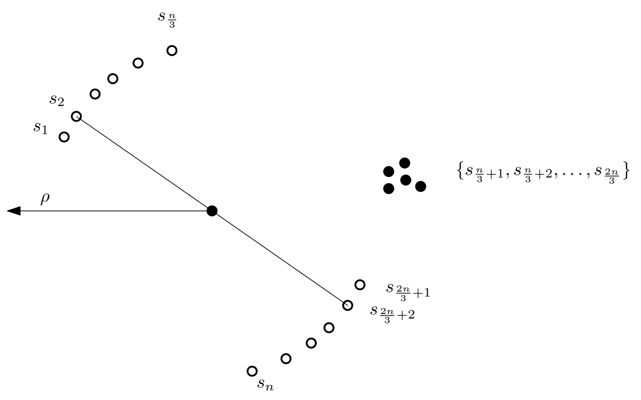

Figure 3.2: Two thirds of the points are arranged on a unit circle around a point c so that $s_{\frac{2n}{3}+i} := s_i$ for all $1 \leq i \leq n/3$. Clearly c is a centerpoint but c doesn't have "high" ray-shooting depth as the ray ρ starting at c intersects at most $n^2/18 + 6n$ edges.

$i \leq n/3$ and a point s_j with $j \geq n - (n/3 - i)$, the edge $\overline{s_i s_j}$ meets the ray ρ. This is because there are at least $n/3$ points to the right of the line ℓ_i. And at least $n - (n/3 - i)$ of them lie below ρ. There are at least $(n/3 - i)$ edges for each s_i with $i \leq n/3$ that meet ρ. Therefore the number of edges meeting ρ is more than $\sum_{i=1}^{n/3}(n/3 - i)$. The claim follows. $\square$

The status in $\mathbb{R}^3$ and higher dimensions

We saw that the optimal bounds on the depth of the Tukey, simplicial, and the RS-median are known in the plane. Unfortunately things are much less well-understood already in $\mathbb{R}^3$. Of the depth measures, as stated earlier, the centerpoint theorem again gives the Tukey depth lower bound of $n/4$ in $\mathbb{R}^3$, and this is optimal.

However, optimal bound for the simplicial depth is not known. It is known that for any set S of n points in $\mathbb{R}^d$, there exists a point lying in $c_d \cdot n^{d+1}$ simplices, where c_d is a constant that depends on the dimension d. The determination of the exact value of c_d is a long-standing open problem in the combinatorial geometry. Bárány [5] proved that $c_d \geq \frac{1}{d!(d+1)^{d+1}}$. Bárány's bound was improved to $\frac{d^2+1}{(d+1)!(d+1)^{d+1}}$ by Wagner [55], who in fact showed that any point of Tukey depth τn is contained in at least the following number of simplices:

$$\frac{(d+1)\tau^d - 2d\tau^{d+1}}{(d+1)!} \cdot n^{d+1} - O(n^d). \tag{3.1}$$

More recently, Bukh, Matoušek, and Nivash [14] gave an elegant construction of a point set S so that no point in $\mathbb{R}^d$ is contained in more than $(n/(d+1))^{d+1}$ simplices defined by S upto lower-order terms. Furthermore, they conjecture that this is the right bound.

For $d = 3$ then, the conjectured bound is $c_3 = 0.0039$. For this case, the bound of Wagner was recently improved by Basit *et al.* [7], where they showed that $c_3 \geq 0.0023$. This was further improved by Gromov [29] using a complicated topological argument; he showed that $c_d \geq 2d/((d+1)(d+1)!^2)$. For $d = 3$ this gives $c_3 \geq 0.0026$. This bound for $\mathbb{R}^3$ has since been improved even further by Mach and Sereni [33] to $c_3 \geq 0.0031$.

In higher dimensions, the notion of RS-depth corresponds to finding a point q such that any ray from q intersects "lots" of $(d-1)$-simplices spanned by points of S. No combinatorial bounds on the RS-depth of such a point are known for $d \geq 3$. It was not studied in the paper that proposed it [26], and their topological technique that was used for the two-dimensional case fails for $d = 3$ and above. Using the bounds on simplicial depth, it is not too hard to derive a first such bound: any set S of n points in $\mathbb{R}^d$ has RS-depth at least $2d/((d+1)((d+1)!)^2) \cdot n^d$. See Theorem 4 in Section 3.4 for a more general bound.

3.3 A Uniform View of data-depth

Given the lack of optimal bounds for simplicial-depth and RS-depth in dimensions higher than two, a leading question is what are the bounds to expect? What would be a good conjecture?

We think that one good way to address such questions is to look at the extreme cases. As it turns out, there is one particular extreme case that is of special interest. Consider the following point set of size n: fix a simplex in $\mathbb{R}^d$ - it is helpful to think of it as a regular simplex albeit this is not a requirement. Place $n/(d+1)$ points on each of its $(d+1)$ vertices. Call such a point set a *simplex-like* point set. For the kind of questions we are considering, this seems to represent an 'extremal' case. In other words, the lower bounds for the data-depth of this point set appear to be true for any point set.

So let's consider the depth measures we have defined in this chapter so far for the simplex-like point set with its points lying on the vertices of a simplex Δ in $\mathbb{R}^d$. Let c be the centroid of Δ, i.e., the arithmetic mean of the $d+1$ vertices of the simplex. Any hyperplane through c has at least one vertex of Δ to each of its two sides. Therefore c has a Tukey depth $\geq n/(d+1)$. And it is best possible because a hyperplane through c exists with d vertices of the simplex on the one side and one vertex on the other side. This meets the exact requirements of the centerpoint theorem. As Δ contains the centroid c, all simplices with a vertex in each of the disjoint sets of points contain c. There are exactly $(n/(d+1))^{d+1}$ such simplices. Finally, any ray from c must intersect at least one facet of S, and so intersect $(n/(d+1))^d$ $(d-1)$-simplices spanned by S and so have that much RS-depth.

Let us give some justification for reliance on this extremal case. First note that, so far, all theoretical and empirical evidence seems to show that as the Tukey depth of a point set increases, so does its simplicial-depth and also its RS-depth. So, for example, for a point set with the Tukey depth of $n/2$, we get the maximum possible values of both simplicial depth and RS-depth. Theoretical results for $\mathbb{R}^2$ relating Tukey depth to simplicial depth have already been discussed in Claim 1.

Second, we outline below an argument showing that, when the Tukey depth of a point set is the lowest possible, i.e., $n/(d+1)$, then we get exactly the bounds one expects from the simplex-like point set. This, together with the first point, leads us to suspect that the bounds derived from the simplex-like point set might indeed be always realizable for any point set.

So consider the following theorem from Boros and Füredi [11]: Given a set S of n points in $\mathbb{R}^d$ with Tukey depth $n/(d+1)$, there exists a point p with depth $n/(d+1)$, and a set $\mathcal{H}$ of $d+1$ halfspaces $\{h_1, \ldots, h_{d+1}\}$, such that (i) $|h_i \cap S| = n/(d+1)$, (ii) p lies on the boundary plane of each h_i, and (iii) $h_1 \cup \ldots \cup h_{d+1}$ cover the entire $\mathbb{R}^d$. It is easy to see that in this configuration, with the given constraints, the $d+1$ regions $A_i = H_i \bigcap \left(\cap_{j \neq i} \overline{H}_j\right)$ each contain exactly $n/(d+1)$ points, and all the other $2^{d+1}-2$ regions are empty. We already assume that the point p has Tukey depth $n/(d+1)$. And it is not too hard to prove that in such a configuration in $\mathbb{R}^d$ it has simplicial depth $(n/(d+1))^{d+1}$ and has RS-depth $(n/(d+1))^d$. In some sense, with respect to the point p, the n points are essentially in a simplex-like position, combinatorially, when Tukey depth is $n/(d+1)$.

Furthermore, the intuition one gets from simplex-like point sets conforms to all the information we have about these problems. It gives exactly the results known for $\mathbb{R}$ (which is trivial) and for $\mathbb{R}^2$. And it matches the conjecture in [14] that there always exists a point of simplicial depth $(n/(d+1))^{d+1}$ (ignoring lower-order terms).

Line-depth in $\mathbb{R}^3$

Let us continue with the consideration of the set S in $\mathbb{R}^3$ where $n/4$ points are placed near each of the vertices of some tetrahedron S. And let c be the centroid of S.

The Tukey depth of c is realized by some half-*space* (a 3-dimensional halfspace) with c lying on its (2-dimensional) boundary. Similarly the RS-depth of c is realized by a ray - a half-*line* (a 1-dimensional halfspace) with c lying on its (0-dimensional) boundary. But this begs the question: what do the half*planes* (2-dimensional halfspaces) with c on their (1-dimensional) boundary define? It is natural to consider the 2-dimensional space defined by a half-plane h with c on its (1-dimensional) boundary and then count the number of *edges* spanned by S that intersect h. What answer is to be expected? Going by the intuition of simplex-like point set, any half-plane through c will intersect at least one edge of S, and so intersect at least $(n/4)^2$ edges spanned by S. Formally, a point $q \in \mathbb{R}^3$ has *line depth* r if any halfplane through q intersects at least r edges spanned by S. The line depth of a point set S is defined as the highest line depth of any point. A half-space in 2-dimensional space is called a half-plane. We conjecture:

Conjecture 1. *Any set S of n points in $\mathbb{R}^3$ has line depth at least $(n/4)^2$.*

Like RS-depth and simplicial-depth in $\mathbb{R}^3$, it seems hard to prove this exact bound using current techniques. But we are able to show the following:

Theorem 3. *Given any set S of n points in $\mathbb{R}^3$, there exists a point q such that any halfplane through q intersects at least $2n^2/49$ edges spanned by S.*

So with the notion of line depth, we have three measures in $\mathbb{R}^3$ for any point q: 2-dimensional space is the familiar Tukey depth, 1-dimensional space gives line depth, and 0-dimensional space gives RS-depth. Intuitively, it is clear that as the dimension of the flat decreases, the degrees of freedom increase and the problem becomes more complicated. On one end, optimal results for the 2-dimensional case (Tukey depth) are known. On the other end, very partial results are known for the 0-dimensional case. It is our hope that the middle 1-dimensional case will be more accessible than the 0-dimensional case. That is another motivation to study the line depth problem.

Proof of Theorem 3. For any set S with n points and Tukey depth τn, our bound is achieved via a two-step strategy. First, we show that there exists an increasing function of τ that lower-bounds the line depth of S. Then, via an alternate technique, we show the existence of a decreasing function of τ that also lower-bounds the line depth of S. Combining the two yields our theorem.

Lemma 3. *Given a set S of n points in $\mathbb{R}^3$, let p be a point with Tukey depth at least τn. Then the line depth of p is at least $(\tau n)^2/2 - o(n)$.*

Proof. Given an arbitrary half-plane H through such a point p, we define a procedure to find edges that intersect H. Starting from H, rotate a half-plane in one direction with the axis of rotation as the bounding line of H. Sort points in S by the order in which they intersect this rotating half-plane, i.e., p_1 is the first point to be hit. Call the half-plane through p_i as H_i, and the plane containing H_i as G_i. Let H_i^+ be the halfspace define by G_i such that $H \subseteq H_i^+$. H partitions H_i^+ into H_i^{++} and H_i^{+-}; denote the wedge containing the points $\{p_1, \dots p_i\}$ as

H_i^{++}. By the definition of Tukey depth of p, $|H_i^{++} \cap S| + |H_i^{+-} \cap S| \geq \tau n$. Observe that for any $i \leq \tau n$, the line segment defined by p_i and $p_j \in H_i^{+-} \cap S$ must intersect H. The number of such line segments can be bounded as

$$T \geq \sum_{k=1}^{\tau n} k = \frac{(\tau n)^2}{2} - o(n). \tag{3.2}$$

□

Note that since $\tau \geq 1/4$ by the Centerpoint Theorem, if you take p to be a Tukey median, Lemma 3 proves the existence of a point with line depth at least $n^2/32$. By a second method, we prove the following lower-bound:

Lemma 4. *Given a set S of n points with Tukey depth τn, there exists a point q with line depth at least $(\tau - 3\tau^2) \cdot n^2$.*

Note that this is a decreasing function of τ for $\tau \in [0.25, 0.5]$. For the proof, we extend the approach in [7] to work for the line depth case.

Given a set S of n points in $\mathbb{R}^3$, with $depth(S) = \tau n$, use Lemma 2 to obtain the point p and a set of four halfspaces $\{h_1, h_2, h_3, h_4\}$ satisfying the stated conditions. We will now show that p gives the required line depth lower-bound of $(\tau - 3\tau^2) \cdot n^2$. As the four halfspaces cover the space, a point in $\mathbb{R}^3$ is in either one, two, or three of the halfspaces. There the four halfspaces partition $\mathbb{R}^3$ into the following convex unbounded regions:

$$A_i = (\bigcap_{l \neq i} \overline{h}_l) \cap h_i, \qquad B_{i,j} = (\bigcap_{l \neq i,j} \overline{h}_l) \cap h_i \cap h_j, \qquad C_i = (\bigcap_{l \neq i} h_l) \cap \overline{h}_i \tag{3.3}$$

See Figure 3.3. The bars at the top of halfspaces indicate the complements, i.e., for $h \subseteq \mathbb{R}^d$, $\overline{h} := \mathbb{R}^d \setminus h$. We note that regions A_i and C_i are *antipodal* around the point p (in the sense that a line through p and intersecting A_i will intersect C_i and not intersect any other region). Similarly region $B_{i,j}$ is antipodal to region $B_{k,l}$ for distinct $1 \leq i, j, k, l \leq 4$. For brevity we also define

$$A := \bigcup_{i \in [4]} A_i, \qquad B := \bigcup_{i,j \in [4], i \neq j} B_{i,j},$$

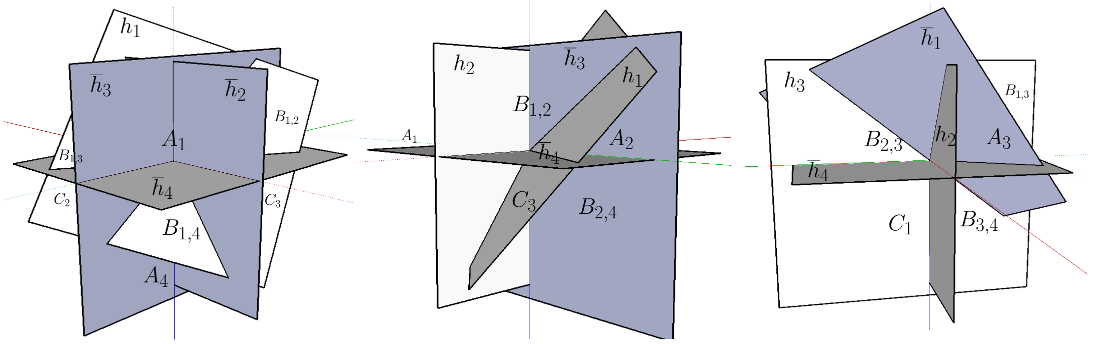

Figure 3.3: Partitioning of $\mathbb{R}^3$ into 14 regions by the four halfspaces as visible from three different angles. The labels on the planes represent the halfspace that is facing the reader.

where $[m]$ denotes the numbers $[m] := \{1, 2, \ldots, m\}$ for an integer m. Set $\alpha_i = |S \cap A_i|/n$, $\beta_{i,j} = |S \cap B_{i,j}|/n$, and $\gamma_i = |S \cap C_i|/n$. Note that we have the following two constraints on the non-negative variables $\alpha_i, \beta_{i,j}$ and γ_i:

$$\tau = \frac{|h_i \cap S|}{n} = \alpha_i + \sum_{j \neq i} \beta_{i,j} + \sum_{j \neq i} \gamma_j \quad \text{for each } i = 1 \ldots 4 \text{ , and} \tag{3.4}$$

$$\sum_i \alpha_i + \sum_{i<j} \beta_{i,j} + \sum_i \gamma_i = 1, \quad \text{as } \{h_1, h_2, h_3, h_4\} \text{ cover } \mathbb{R}^3 \setminus \{p\}. \tag{3.5}$$

Summing up (3.4) for all four halfspaces, and subtracting (3.5) from it, we get

$$\sum_{i<j} \beta_{i,j} + 2 \cdot \sum_i \gamma_i = 4 \cdot \tau - 1. \tag{3.6}$$

Therefore, $\sum_{i<j} \beta_{i,j} + \sum_i \gamma_i \leq 4\tau - 1$. This fact, together with equation (3.4), implies that $1 - 3\tau \leq \alpha_i \leq \tau$ for $i = 1 \ldots 4$.

In what follows we need the following lemma.

Lemma 5. *For any* $1 \leq i < j \leq 4$, *we have* $\alpha_i\alpha_j + \beta_{i,j}\alpha_i \geq \tau - 3\tau^2$. *Similarly,* $\alpha_i\alpha_j + \beta_{i,j}\alpha_j \geq \tau - 3\tau^2$.

Proof. Assume the other two halfspaces are h_k and h_l (other than h_i and h_j). Since $|S \cap \overline{h}_k| =$

$1-\tau$, and $|S \cap h_l| = \tau$, we get $S \cap (\overline{h}_i \setminus h_l) = \alpha_i + \beta_{i,j} + \alpha_j \geq 1 - 2\tau$. Then we have

$$\alpha_i\alpha_j + \beta_{i,j}\alpha_i \geq \alpha_i\alpha_j + (1 - 2\tau - \alpha_i - \alpha_j)\alpha_i = \alpha_i(1 - 2\tau - \alpha_i).$$

This last term is minimized at the extreme values of α_i, which are either τ or $1 - 3\tau$, both yielding a lower-bound of $\tau(1 - 3\tau)$. $\square$

We will repeatedly use the following fact in different cases below to count the number of line segments intersecting an arbitrary half plane H.

Fact 2. *Given a set of halfspaces $\mathcal{H}$ in $\mathbb{R}^3$, let X be the convex region of their common intersection, and let H be any set such that $X \setminus H$ has more than one path-connected component. Then if p and q are points in two different components, the edge $\overline{pq}$ must intersect H.*

Let A_i^+, A_i^- be a partition of A_i and $B_{i,j}^+$, $B_{i,j}^-$ a partition of $B_{i,j}$. Then define: $\alpha_i^+ = \frac{|A_i^+ \cap S|}{n}, \alpha_i^- = \frac{|A_i^- \cap S|}{n}, \beta_{i,j}^+ = \frac{|B_{i,j}^+ \cap S|}{n}, \beta_{i,j}^- = \frac{|B_{i,j}^- \cap S|}{n}$.

Claim 2. $\alpha_i \cdot \alpha_j + \beta_{i,j}^+ \cdot \alpha_j + \beta_{i,j}^- \cdot \alpha_i \geq \tau - 3\tau^2$.

Proof.

$$\begin{aligned}
\alpha_i \cdot \alpha_j + \beta_{i,j}^+ \cdot \alpha_j + \beta_{i,j}^- \cdot \alpha_i &\geq \alpha_i\alpha_j + \min(\alpha_i, \alpha_j)(\beta_{i,j}^+ + \beta_{i,j}^-) \\
&= \alpha_i\alpha_j + \beta_{i,j}\min(\alpha_i, \alpha_j) \\
&= \min(\alpha_i\alpha_j + \beta_{i,j}\alpha_i, \alpha_i\alpha_j + \beta_{i,j}\alpha_j) \geq \tau - 3\tau^2
\end{aligned}$$

by Lemma 5. $\square$

Claim 3. $\alpha_i^+ \cdot \alpha_j + \alpha_i^- \cdot \alpha_k + \alpha_i^+ \cdot \beta_{i,j}^- + \beta_{i,j}^+ \cdot \alpha_j + \alpha_i^- \cdot \beta_{i,k} \geq \tau - 3\tau^2$.

Proof. The left-hand side is equal to

$$\begin{aligned}
&= \alpha_i^+(\alpha_j + \beta_{i,j}^-) + \alpha_i^-(\alpha_k + \beta_{i,k}) + \alpha_j\beta_{i,j}^+ \\
&\geq \min(\alpha_j + \beta_{i,j}^-, \alpha_k + \beta_{i,k})(\alpha_i^+ + \alpha_i^-) + \alpha_j\beta_{i,j}^+ \\
&= \alpha_i \min(\alpha_j + \beta_{i,j}^-, \alpha_k + \beta_{i,k}) + \alpha_j\beta_{i,j}^+ \\
&= \min(\alpha_i\alpha_j + \alpha_i\beta_{i,j}^- + \alpha_j\beta_{i,j}^+, \alpha_i\alpha_k + \beta_{i,k}\alpha_i + \alpha_j\beta_{i,j}^+) \\
&\geq \min(\alpha_i\alpha_j + \min(\alpha_i, \alpha_j)(\beta_{i,j}^+ + \beta_{i,j}^-), \alpha_i\alpha_k + \beta_{i,k}\alpha_i) \\
&= \min(\min(\alpha_i\alpha_j + \alpha_i\beta_{i,j}, \alpha_i\alpha_j + \alpha_j\beta_{i,j}), \alpha_i\alpha_k + \beta_{i,k}\alpha_i) \\
&= \min(\alpha_i\alpha_j + \alpha_i\beta_{i,j}, \alpha_i\alpha_j + \alpha_j\beta_{i,j}, \alpha_i\alpha_k + \beta_{i,k}\alpha_i) \\
&\geq \tau - 3\tau^2,
\end{aligned}$$

where the last inequality follows from Lemma 5. □

Let p be as defined in Lemma 2; we will show that any half-plane through p intersects at least $(\tau - 3\tau^2) \cdot n^2$ edges spanned by S. We identify a line ℓ as the supporting line of the halfplane H if it lies on the boundary of H. The supporting line ℓ of H passes through a pair of antipodal regions among the fourteen regions define in Equation 3.3 as well as the point p.

There are two obvious possibilities for a halfplane H: either it intersects A (recall that A is the union of $A_i, i \in [4]$) or it does not intersect A (**Case 2** below). For the first possibility we can have three possible sub cases: the supporting line ℓ of H passes through some A_i but H doesn't intersect $A \setminus A_i$ (**Case 1**) or H intersects A but the supporting line ℓ doesn't (**Case 4**) or ℓ passes through some A_i and H intersects some $A_j, j \neq i$ (**Case 3**).

Let $\eta(H)$ denote the number of edges meeting halfplane H. It suffices to prove the bound on $\eta(H)$ separately for the following four cases. We will use terms *above (or below) a half-plane H* to distinguish between points that lie on opposite sides of the plane passing through H. Also we call a pair of regions as neighbors (to each other) if exactly one halfspace h_i separates points in one region from the points in the other region.

Case 1: Supporting line ℓ passes through A_i and H does not intersect $A \setminus A_i$ for $i \in [4]$

Note that in this case the supporting line ℓ also passes through C_i. By definition $C_i = (\bigcap_{j \neq i} h_j) \cap \overline{h}_i$. Since there is no point common in all four (open) halfspaces (or their complements) we also have that $C_i = (\bigcap_{j \neq i} h_j)$. There are three neighbors of C_i: $B_{j,k}, B_{k,l}, B_{j,l}$ where $j, k, l \in [4] \setminus \{i\}$ and any halfplane H with supporting line passing through C_i will intersect exactly one of them (see Figure 3.4). Without loss of generality let us assume that H intersects some $B_{j,k}$ where $j, k \neq i$, and let $\mathcal{H} = \{\overline{h}_i, \overline{h}_l\}$. By definition, all the points in $A_j \cup B_{j,k} \cup A_k$ lie in the intersection of halfspaces in $\mathcal{H}$. As H does not intersect A_j and A_k (by assumption), H partitions the region defined by the intersection of halfspaces in $\mathcal{H}$ into two pieces, with $A_j \cup B^+_{j,k}$ lying above H, while $A_k \cup B^-_{j,k}$ lies below H. Using Fact 2 with $\mathcal{H}$ and H, we see that the following number of edges must intersect H: $\eta(H) \geq \alpha_j \cdot \alpha_k + \beta^+_{j,k} \cdot \alpha_k + \beta^-_{j,k} \cdot \alpha_j$. From Claim 2, it follows that $\eta(H) \geq \tau - 3\tau^2$.

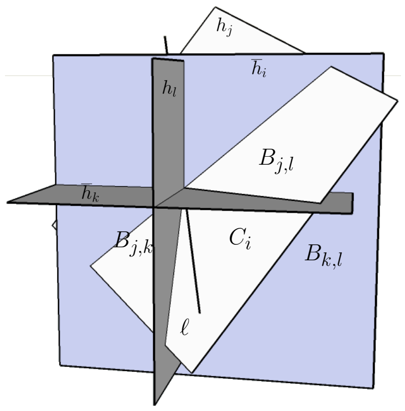

Figure 3.4: Region C_i has three neighboring regions: $B_{j,k}, B_{j,l}$ and $B_{k,l}$ and for a halfplane H with supporting line ℓ passing through A_i and C_i, H intersects exactly one of them.

Case 2: (H does not intersect any A_i for $i \in [4]$).

If H does not intersect any A_i then it must be that the supporting line ℓ passes through $B_{i,j}$ and $B_{k,l}$ and H intersects B_{xy} with $x \in \{i, j\}$ and $y \in \{k, l\}$. With loss of generality assume that

$x = j$ and $y = k$. Let $\mathcal{H} = \{\overline{h}_i, \overline{h}_l\}$. Then, by definition, all the points in $A_j \cup B_{j,k} \cup A_k$ lie in the intersection of halfspaces in $\mathcal{H}$. And in this case, H partitions the region defined by the intersection of halfspaces in $\mathcal{H}$ into two pieces, with $A_j \cup B^+_{j,k}$ lying above H, and $A_k \cup B^-_{j,k}$ lying below H. Using Fact 2 with $\mathcal{H}$ and H, we get : $\eta(H) \geq \alpha_j \cdot \alpha_k + \beta^+_{j,k} \cdot \alpha_k + \beta^-_{j,k} \cdot \alpha_j$. From Claim 2, it follows that $\eta(H) \geq \tau - 3\tau^2$.

Case 3: (Supporting line ℓ passes through A_i and H also intersects some A_j for some $i, j \in [4], j \neq i$).

In this case H may also intersect $B_{j,k}$ or $B_{j,l}$ but not both. Without loss of generality H intersects $B_{j,k}$ (if H does not intersect any of them then we can take either one part of the assumed partition to contain no points of S). Let $\mathcal{H} = \{\overline{h}_i, \overline{h}_l\}$. Then, by definition, all the points in $A_j \cup B_{j,k} \cup A_k$ lie in the intersection of halfspaces in $\mathcal{H}$. And in this case, H partitions the region defined by the intersection of halfspaces in $\mathcal{H}$ into two pieces, with $A^+_j \cup B^+_{j,k}$ lying above H, and $A^-_j \cup A_k \cup B^-_{j,k}$ lying below H. Using Fact 2 with $\mathcal{H}$ and H, the edges with endpoints in the following pairs of regions are intersecting H: $(A^+_j, A_k), (A^+_j, B^-_{j,k}), (A_k, B^+_{j,k})$. Similarly H also partitions $\overline{h}_i \cap \overline{h}_k$; setting $\mathcal{H} = \{\overline{h}_i, \overline{h}_k\}$, the edges with endpoints in following pairs must intersect H: (A_l, A^-_j), $(A^-_j, B_{j,l})$. Therefore, $\eta(H) \geq \alpha^+_j \cdot \alpha_k + \alpha^-_j \cdot \alpha_l + \alpha^+_j \cdot \beta^-_{j,k} + \beta^+_{j,k} \cdot \alpha_k + \alpha^-_j \cdot \beta_{j,l}$. From Claim 3, it follows that $\eta(H) \geq \tau - 3\tau^2$.

Case 4: (H intersects A but the supporting line ℓ doesn't).

Since supporting line ℓ does not intersect A, it must intersect B. Without loss of any generality, ℓ passes through $B_{i,j}$ and $B_{k,l}$ and H intersects A_j. In this case H may also intersect $B_{j,k}$ or $B_{j,l}$ but not both. Again without loss of generality H intersects $B_{j,k}$ (if H does not intersect any of them then we can take either one part of the assumed partition to contain no points of S). Let $\mathcal{H} = \{\overline{h}_i, \overline{h}_l\}$. Then, by definition, all the points in $A_j \cup B_{j,k} \cup A_k$ lie in the intersection of halfspaces in $\mathcal{H}$. And in this case of H, H partitions the region defined by the intersection

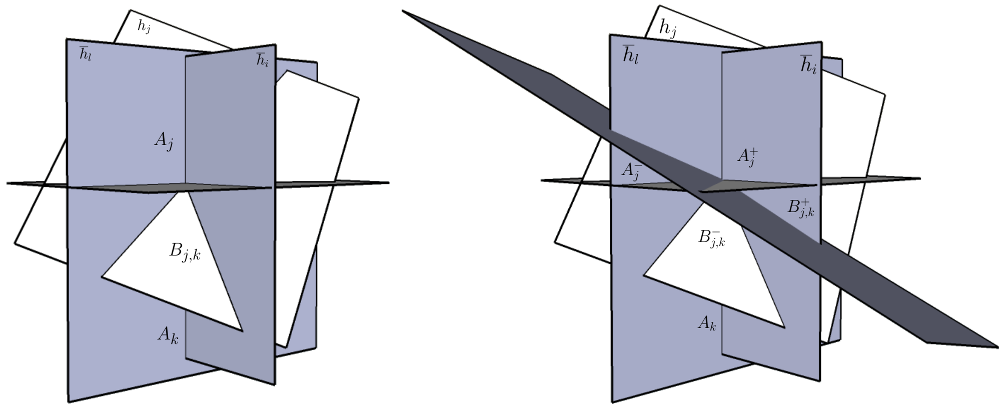

Figure 3.5: Halfplane H partitions $\overline{h}_i \cap \overline{h}_l = A_j \cup B_{j,k} \cup A_k$ into two parts: $A_j^+ \cup B_{j,k}^+$ lying above H and $A_j^- \cup A_k \cup B_{j,k}^-$ lying below H.

of halfspaces in $\mathcal{H}$ into two pieces, with $A_j^+ \cup B_{j,k}^+$ lying above H, and $A_j^- \cup A_k \cup B_{j,k}^-$ lying below H as in Figure 3.5. Using Fact 2 with $\mathcal{H}$ and H, we get that edges with endpoints in the following pairs of regions are intersecting H: $(A_j^+, A_k), (A_j^+, B_{j,k}^-), (A_k, B_{j,k}^+)$. Similarly H also partitions $\overline{h}_i \cap \overline{h}_k$; setting $\mathcal{H} = \{\overline{h}_i, \overline{h}_k\}$, the edges with endpoints in following pairs must intersect H: $(A_l, A_j^-), (A_j^-, B_{j,l})$. Therefore, the following number of edges must intersect H: $\eta(H) \geq \alpha_j^+ \cdot \alpha_k + \alpha_j^- \cdot \alpha_l + \alpha_j^+ \cdot \beta_{j,k}^- + \beta_{j,k}^+ \cdot \alpha_k + \alpha_j^- \cdot \beta_{j,l}$. From Claim 3, it follows that $\eta(H) \geq \tau - 3\tau^2$. This completes the proof of Lemma 4.

□

Finally, to complete the proof of Theorem 3: take any set S of n points with Tukey depth τn. If $\tau \geq 0.285$, Lemma 3 gives a point with line depth at least $(0.285)^2 n^2/2 \geq n^2/24.5$. On the other hand, if $\tau < 0.285$, Lemma 4 gives a point with line depth at least $(0.285 - 3(0.285)^2)n^2 \geq n^2/24.5$. □

3.3.1 Relations between data-depth measures in $\mathbb{R}^3$

For $d = 2$, we have two fundamental measures: Tukey depth, and RS-depth. Recall from section 3.2.3 that any point of RS-depth $n^2/9$ has Tukey depth at least $n/3$. However the converse is not necessarily true. So there is a hierarchy in dimension 2. Is the analogous property true in $\mathbb{R}^3$?

Speculation 1. *Let q be a point in $\mathbb{R}^3$ with RS-depth $(n/4)^3$. Then q has line depth at least $(n/4)^2$. Similarly, let q be a point with line depth $(n/4)^2$. Then q has Tukey depth at least $(n/4)$.*

Unfortunately, the hierarchical structure that is present in $\mathbb{R}^2$ is absent in $\mathbb{R}^3$.

Lemma 6. *There exists a point set S and a point q both in $\mathbb{R}^3$ such that the line depth of q is at least cn^2 with $c > \frac{1}{16}$ while the Tukey depth of q is at most $n/4$ where n is large enough.*

Proof. To prove this statement we will construct one such set of n points and a point q with claimed bounds on its line depth and Tukey depth. Our point set S has two parts A and B with $|A| = 3n/4$ and $|B| = n/4$. We place the points in $A = \{a_1, a_2, \ldots, a_{3n/4}\}$ around the origin at regular distance in the unit radius circle in the xy-plane with $z = 0$, i.e., the point $a_i := (\cos(\frac{4i}{3n}2\pi), \sin(\frac{4i}{3n}2\pi), 0)$. Similarly the points in B lie in a circle of small enough radius $\epsilon > 0$ around the point $(0, 0, 1)$ at regular distance again in the xy-plane with $z = 1$. Any $\epsilon < \frac{2\pi}{3n/4}$ will work but for simplicity assume ϵ is arbitrarily small. We fix q to be the midpoint on the line segment between the centers of these two circles i.e. $q = (0, 0, 1/2)$. It is easy to see that the Tukey depth of q is at most $n/4$.

We want to show that for any halfplane H through q there are more than $n^2/16$ edges incident on the pairs of points in S that intersect H. We identify a line ℓ as the supporting line of the halfplane H if it lies on the boundary of H. The supporting line ℓ of H passes through the point q. By symmetry of points in A and B we can assume, without loss of any generality, that the line ℓ lies on xz-plane and that the halfplane H lies in the positive y halfspace. For a point u in the three-dimensional space we use $x(u), y(u), z(u)$ to represent $x, y,$ and z coordinates of u respectively. We write $Conv(A)$ and $Conv(B)$ for respective convex hulls of the points in A and B. For a set Q of points $\pi(Q)$ denotes the plane through Q whenever Q determines a unique such plane. Unless a halfplane misses both the convex hulls (which is described in

Case 1), its supporting line ℓ may or may not intersect $Conv(B)$. The scenario when it does intersect $Conv(B)$ is described in Case 2 below. We divide the case when it does not intersect $Conv(B)$ into further sub cases based on whether ℓ intersect $Conv(A)$ or not.

Case 1: H misses both $Conv(A)$ and $Conv(B)$.

It is easy to see that for all $a_i = (u, v, 0) \in A$ with $v \geq 0$ and for all $b_j \in B$ edge $\overline{a_i b_j}$ meets H. We have

$$\begin{aligned} \eta(H) &\geq \frac{n}{4} \times \frac{3n}{4} \times \frac{1}{2} \\ &= \frac{3n^2}{32} > \frac{n^2}{16}. \end{aligned}$$

As a matter of fact, the number of the edges intersecting H is exactly $\frac{3n^2}{32}$ in this case.

Case 2: The supporting line ℓ passes through $Conv(B)$.

Since the points in B are placed in a small enough circle the line ℓ passes very close to the origin i.e. the euclidean distance between origin and the closest point on ℓ is at most ϵ. Among all the edges that are incident on a pair of vertices in A only a linear (in $|A|$) number of them pass through any circle of radius ϵ around the origin . That is because for each point a_i in A there are at most 4 points a' such that the edges of the form $\overline{a_i a'}$ intersect such a circle due to the choice of ϵ to be an arbitrarily small number. Since a halfplane passing through q and the origin is intersected by $(1/2)(3n/8)(3n/8)$ edges with both endpoints in A,

$$\begin{aligned} \eta(H) &\geq \frac{3n}{8} \times \frac{3n}{8} \times \frac{1}{2} - cn \\ &= \frac{9n^2}{128} - cn > \frac{n^2}{16}, \end{aligned}$$

where cn is the linear number of edges that pass through a circle of radius ϵ around the origin.

For the rest of the proof we will assume that the supporting line ℓ does not pass through $Conv(B)$. In Case 3 and Case 4 we describe the scenario when ℓ does not intersect $Conv(A)$

either. And then for Case 5 and Case 6 we assume that the supporting line ℓ does pass through $Conv(A)$.

Case 3: H intersects $Conv(A)$ but the supporting line ℓ doesn't.

Note that H can't intersect $Conv(B)$ in this case. Let H partition the points in A into two parts, A_1 lying above and A_2 lying below H. All edges with one endpoint in A_1 and the other endpoint in A_2 meet H. Similarly all edges with one point in B and other in A_2 with the positive y-coordinate meet H too. If there are k points in A_1 then we can assume that $0 \le k \le \frac{3n}{8}$. We have

$$\begin{aligned}\eta(H) &\geq k \times (\frac{3n}{4} - k) + (\frac{3n}{8} - k) \times \frac{n}{4} \\ &= \frac{3n^2}{32} + \frac{2nk}{4} - k^2 \\ &= \frac{3n^2}{32} + k(n/2 - k) > \frac{n^2}{16}.\end{aligned}$$

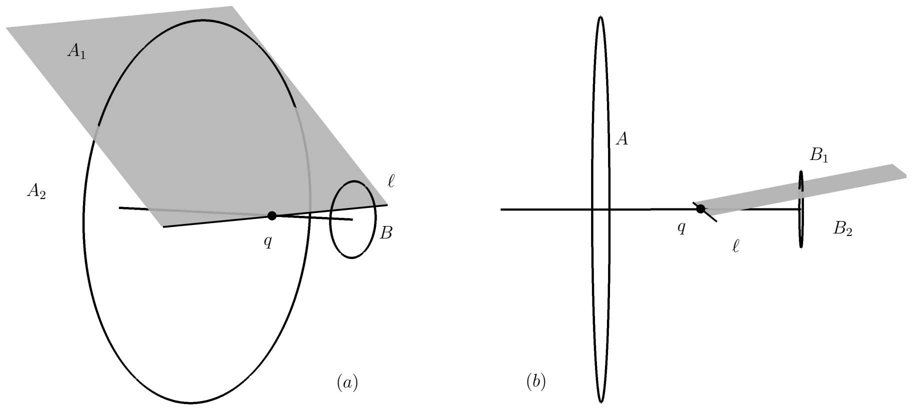

Figure 3.6: Figure(a) on the left illustrates Case 3 when H partitions A into A_1 and A_2. Similarly figure(b) on the right illustrates Case 4 when H partitions B into B_1 and B_2.

Case 4: H does not intersect $Conv(A)$, intersects $Conv(B)$, and the supporting line ℓ doesn't intersect $Conv(B)$.

Let H partition the points in B into two parts B_1 lying above and B_2 lying below H. All the edges with one point in B_1 and other in B_2 meet H. Similarly all edges with one point in B_1 and other in A with a negative y-coordinate meet H. And all the edges with one point in B_2 and other in A with a positive y-coordinate meet H as well. If there are k points in B_1 then we can assume that $0 \leq k \leq \frac{n}{8}$. We have

$$\begin{aligned}\eta(H) &\geq k \times \frac{n}{4} - k + k \times \frac{3n}{8} + (\frac{n}{4} - k) \times \frac{3n}{8} \\ &= \frac{3n^2}{32} k \times \frac{n}{4} - k > \frac{n^2}{16}\end{aligned}$$

Case 5: The supporting line ℓ intersects $Conv(A)$, but H doesn't intersect $Conv(B)$.

H partitions the points in A that lie in positive y halfspace into two sets A_1 and A_2 such that the points in A_1 and the points in B lie to the left of the plane $\pi(H)$ while the points in A_2 lie to the right. Let $|A_1| = k$ and $|A_2| = 3n/8 - k$. When $k \leq n/8$ there are at least $n/4 \times 2n/8$ edges with one endpoint in B and the other in A_2 and all such edges intersect H. Similarly when $n/8 < k < 2n/8$ there are at least $n/8 \times 2n/8$ edges with one endpoint in A_1 and the other in A_2 and $n/4 \times n/8$ edges with one endpoint in A_1 and the other endpoint in B - again all such edges intersect H. We have a slightly more interesting case when $k \geq 2n/8$. If there are $3n/8 - k'$ points in A above $\pi(H)$ and $3n/8 + k'$ points below then we know that there are at least $(3n/8 - k') \times (3n/8 + k')/2$ edges that intersect H where $k' \leq k \leq n/8$. The number of edges meeting H is at least $\frac{n^2}{16}$ in all cases.

Here is the final case:

Case 6: The supporting line ℓ intersects $Conv(A)$, H intersects $Conv(B)$, but ℓ does not intersect $Conv(B)$.

Let $p_0 = (u_0, v_0, 0)$ and $p_1 = (u_1, v_1, 1)$ be the points where ℓ intersects the plane $z = 0$ and the plane $z = 1$ respectively. By the symmetry of the points in A and B we may assume that $u_0 < 0$. Since ℓ passes through the point q at $z = 1/2$, this implies that $u_1 > 0$. Also notice that as H intersects $Conv(B)$, H must intersect the plane $z = 0$ only in the negative X half i.e. i.e $x(H \cap \{p \in \mathbb{R}^3 : z(p) = 0\}) < 0$. Let $A' := A \cap \{(x, y, z) : y \geq 0\}$ i.e. the points in A that lie in the positive y halfspace. H partitions the points in A' into two sets: A_1 lying below H and A_2 lying above. Similarly H partitions B into B_1 lying below H and B_2 lying above. Let $|A_1| = k$ and $|A_2| = 3n/8 - k$. Due the small choice of ϵ, the radius of circle that contains B, it must be that $k \leq 2$. Also with $|B_1| = j$ and $|B_2| = n/4 - j$, it must be that $j > n/8$. All edges with one endpoint in $A_1 \cup B_1$ and the other endpoint in $A_2 \cup B_2$ meet H. Also all edges with one endpoint in B_2 and the other endpoint in $A \setminus A'$ meet H. Therefore,

$$\begin{aligned}
\eta(H) &\geq j \times (\frac{3n}{8} - k) + (\frac{3n}{8} + k) \times (\frac{n}{4} - j) \\
&> (\frac{3n}{8} - k) \times (\frac{n}{4}) \\
&= \frac{3n^2}{32} - \frac{2n}{4} > \frac{n^2}{16}
\end{aligned}$$

as required. $\square$

In summary then, unlike the $d = 2$ case where a point of high RS-depth has a correspondingly high Tukey depth, the three measures in $\mathbb{R}^3$ do not have any such hierarchical relation.

3.4 k-Centerpoints Conjectures

The previous view can be extended to d-dimensions, giving the following "affine k-centerpoints" conjectures:

Conjecture 2 (Affine k-Centerpoints Conjectures). *Given a set S of n points in $\mathbb{R}^d$, and an* *integer* $0 \leq k \leq d-1$*, there exists a point* $q \in \mathbb{R}^d$ *such that any* $(d-k)$*-half flat through* q *intersects at least* $(n/(d+1))^{k+1}$ k*-simplices spanned by* S*.*

The case $k = 0$ is the centerpoint theorem in any $\mathbb{R}^d$. The case $d = 2, k = 1$ is the RS-depth result of [26].

The case $d = 3, k = 1$ is the line depth theorem, for which we have presented Theorem 3.

For the general k-centerpoints problems, it is not hard to see that for every k and d there exists a point q and a constant $c_{d,k}$ such that any $(d-k)$-half flat through q intersects at least $c_{d,k} \cdot n^{k+1}$ k-simplices spanned by S. The following bounds were proved in [48] by extending the technique of Bárány [5] and using the result of Gromov [29].

Theorem 4. *Given a set* S *of* n *points in* $\mathbb{R}^d$*, and an integer* $0 \leq k \leq d-1$*, there exists a point* $q \in \mathbb{R}^d$ *such that any* $(d-k)$*-half flat through* q *intersects at least*

$$\max\left\{ \binom{n/(d+1)}{k+1},\quad \frac{2d}{(d+1)(d+1)!\binom{n}{d-k}} \cdot \binom{n}{d+1} \right\}$$

k*-simplices spanned by* S*.*

Using some approximations for binomial coefficients and factorials, it can be showed that first bound dominates the expression above for $k < 0.9d$ and for $k > 0.9d$ the second bound starts to dominate for $d > 800$. For smaller values of d, this threshold tends to grow with d from $0.8d$ to $0.9d$.

Coming back to the simplex-like point set, one can further observe another thing. Let S denote the tetrahedron of the simplex-like point set in $\mathbb{R}^3$ and suppose we're studying the line depth of the centroid c. Then, as mentioned earlier, any half-plane through c intersects at least one edge of the tetrahedron, so it intersects at least $(n/4)^2$ edges spanned by S. But, in fact, something stronger is true: take any line l through c and move l in any way to 'infinity' (i.e., outside the convex-hull of S). Then it has to cross at least one edge of the tetrahedron. So the property for the simplex-like point set is in fact topological in nature.

In fact, this property is already true for centerpoints: if a point q has Tukey depth r, then any plane through q intersects at least r points as we move the plane to infinity, regardless of whether the movement is any arbitrary continuous movement or affine. See [3] for a discussion of this.

The "affine" k-centerpoints conjectures can therefore be strengthened to a more natural topological version:

Conjecture 3 (k-Centerpoints Conjectures)**.** *Given a set S of n points in $\mathbb{R}^d$, and an integer $0 \leq k \leq d-1$, there exists a point $q \in \mathbb{R}^d$ such that any $(d-k-1)$-flat through q must cross at least $(n/(d+1))^{k+1}$ k-simplices spanned by S to move the flat until it leaves $Conv(S)$.*

We now give a proof of these topological k-centerpoints conjectures in $\mathbb{R}^2$.

Theorem 5. *For any set S of n points in $\mathbb{R}^2$, the k-centerpoints conjectures are true.*

Proof. As centerpoint-depth is already proven to be topological, we only have to resolve the RS-depth case.

We will actually prove the contrapositive: given a set S of n points in $\mathbb{R}^2$, let q be the point with RS-depth ρ. Take any curve γ from q to a point at ∞ that intersects at most ρ edges spanned by S. We want to show that there exists a ray from q that also intersects at most ρ edges spanned by S. We prove this in two steps: first by replacing γ by a piecewise linear curve, and then replacing this piecewise linear curve with a ray.

Given any curve γ, let $\eta(\gamma)$ denote the number of edges spanned by S that γ intersects. We assume that $\eta(\gamma)$ is finite.

Lemma 7. *Given a curve γ with one endpoint at q and the other endpoint at infinity, there exists a piecewise linear curve γ' starting at q and ending at a point at infinity, such that $\eta(\gamma) = \eta(\gamma')$.*

Proof. Given S, let $\mathcal{A}$ be the arrangement induced by $\Theta(n^2)$ lines supporting all the edges spanned by points of S. The curve γ enters and leaves a number of cells in the arrangement

$\mathcal{A}$. Lets say it enters some cell C at point s_i and then leaves that cell at point e_i. Replace this portion of γ between s_i and e_i by the straight-line edge s_ie_i (by convexity of C, this lies completely inside C). Note that γ cannot intersect any edge in the interior of any cell C (no edge can intersect a cell of this arrangement). Therefore, the RS-depth does not change by this replacement.

Repeating this for each cell that γ enters and leaves, we get a sequence $\langle q = s_1, \ldots, s_m, u_1\rangle$, $s_i \in \mathbb{R}^2, u_1 \in \mathbb{S}^1$, which represents the piecewise linear curve γ' defined by the segments $s_1s_2, \ldots, s_{m-1}s_m$ together with the half-infinite ray $\vec{s}_m$ starting from s_m in the direction u_1. Finally, as discussed above, we have $\eta(\gamma) = \eta(\gamma')$. See Figure 3.7(a). □

Let γ' be the piecewise linear curve defined by the sequence $\langle s_1, \ldots, s_m, u_1\rangle$ as above. Consider the one-bend curve γ'' starting at s_{m-1} and defined by the segment $s_{m-1}s_m$ together with the half-infinite ray $\vec{s}_m$ in direction u_1. The Lemma below shows that there exists a direction $u_2 \in \mathbb{S}^1$ such that the half-infinite ray r starting at s_{m-1} in direction u_2 has $\eta(r) \leq \eta(\gamma'')$. In other words, γ'' can be 'straightened' to a ray r without increasing the number of edges intersected. This implies that

$$\eta(\langle s_1, \ldots, s_{m-1}, u_2\rangle) \leq \eta(\langle s_1, \ldots, s_m, u_1\rangle)$$

We now repeat this for the curve defined by $\langle s_1, \ldots, s_{m-1}, u_2\rangle$ to get another curve with one less bend. And so on till we get a ray r starting at q in direction $u \in \mathbb{S}^1$. By induction,

$$\eta(\langle q = s_1, u\rangle) \leq \eta(\langle s_1, s_2, u_{m-1}\rangle) \leq \eta(\langle s_1, s_2, s_3, u_{m-2}\rangle) \leq \cdots \leq \eta(\langle s_1, \ldots, s_m, u_1\rangle)$$

and the proof of Theorem 5 is completed. It remains to prove the following.

Lemma 8. *Given a piecewise linear curve γ' defined by $\langle q_1, q_2, u_2\rangle$, there exists a direction $u_1 \in \mathbb{S}^1$ such that $\eta(\langle q_1, u_1\rangle) \leq \eta(\langle q_1, q_2, u_2\rangle)$.*

Proof. See Figure 3.7(b). First we show the following:

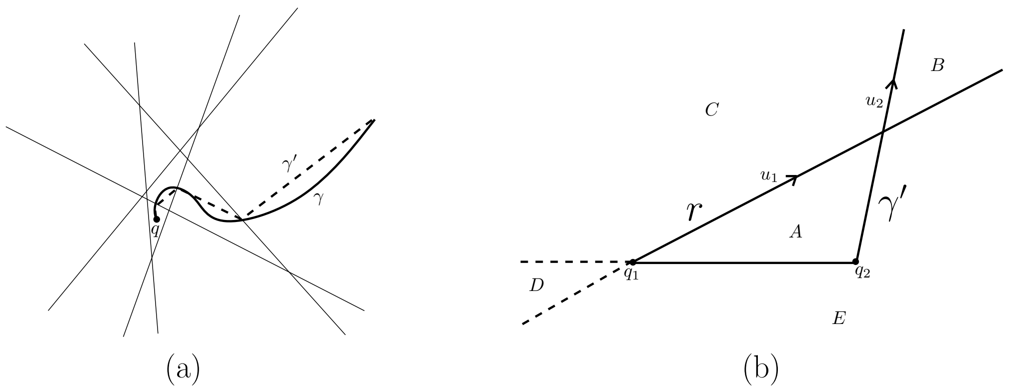

Figure 3.7: (a) Converting γ to piecewise linear, (b) Illustration for Lemma 8.

Claim 4. *There exists a direction $u_1 \in \mathbb{S}^1$ such that:*

(i) the ray from q_1 in the direction u_1 intersects the ray from q_2 in direction u_2, (ii) number of points in region A is equal to the number of points in region B (Figure 3.7(b)).

Proof. Start with the ray r from q_1 through q_2. And continuously rotate r towards the direction u_2. Initially the region A is empty while B may contain some of the data points. When r points in the direction u_2 the opposite is true. At each event when r meets a data point either the number of data points in A increase by 1 or the number of data points in B decrease by 1 *after* the event. There must be a ray from q_1 in a direction between q_2 and u_2 that has equal number of points in both the regions A and B (which could be 0). $\square$

Let r be this ray from q_1 in direction u_1. And assume the regions A and B each contain t points. We now prove that this is the required ray by showing that the number of edges intersected by r is at most those intersected by the curve γ'.

Edges that intersect both r and γ' or intersect neither, contribute equally, and so they can be ignored. Consider the remaining edges spanned by S.

Observation 1. *Any edge $e = \{p_i, p_j\}$ that intersects r (resp. γ'), but not γ' (resp. r), must have exactly one endpoint in region A or B.*

Proof. We prove the contrapositive. If e has one endpoint in A and the other in B, then it must cross both r and γ'. On the other hand, if no endpoint of e is in A or B, then either e intersects both, or neither. $\square$

Therefore consider each point p_i not in A or in B (see Figure 3.7(b)):

- $p_i \in C$. Then p_i has exactly t edges (to points in B) intersecting γ' , and exactly t edges (to points in A) intersecting r .
- $p_i \in D$. Then p_i has at least t edges (to points in B and possibly in A) intersecting γ' and at most t edges (to points in A) intersecting r .
- $p_i \in E$. Then p_i has at least t edges (to points in A and possibly in B) intersecting γ' and exactly t edges (to points in B) intersecting r .

Summing up over all p_i proves the Lemma. $\square$

Lemma 8 completes the proof of Theorem 5. $\square$

3.5 Conclusion

In this chapter we introduced a new data-depth measure, the *line depth* and proved the first bounds on the depth of a median with respect to line depth. We saw that line depth acts as a conceptual bridge between other well-known data-depth measures and suggests that a fundamental relation exists. This was formalized in Conjecture 3. We observe the following hierarchy among the data-depth measures in two dimensions: a point of “high” RS-depth has “high” Tukey and simplicial depth. Lemma 6 shows that a similar hierarchy among the data-depth measures in $\mathbb{R}^3$ doesn’t exist.

This chapter concludes many open questions. There is still a gap between the upper and lower bounds in the First Selection Lemma in $\mathbb{R}^3$. Also the upper bound proved in Theorem 3 for the line depth median is fat from $n^2/16$ the lower bound suggested in Conjecture 3. Furthermore there are no nontrivial lower bounds on the depth of ray-shooting median in three dimensions except for the ones suggested by Theorem 4.

In the next chapter we will focus on the computational problem of computing a point of “high” ray-shooting depth in $\mathbb{R}^2$.

Chapter 4

Algorithms for Ray-Shooting Depth

4.1 Introduction

In the previous chapter, we discussed the notion of data-depth measures with respect to their combinatorial properties. Another important criterion for the utility of a data-depth measure is the time complexity requirement for its computation. In this chapter we focus on algorithmic aspects. The new results are for the most recent depth notion, *ray-shooting depth* where little work has been done so far on the computational and algorithmic aspects.

We begin by summarizing some computational facts known for Tukey depth and for simplicial depth. Let S be a set of n given data points in $\mathbb{R}_1$. By the Centerpoint Theorem there is always a point in $\mathbb{R}^1$ of Tukey depth at least $n/2$ and this is in fact the maximal possible Tukey depth. In this case the depth of S is $\lceil(n+1)/2\rceil$, the depth of a median. Its somewhat surprising that before about 1975 it was not known if a median could be found faster than by sorting the elements of S and then returning a "middle" value - i.e., in time $O(n \log n)$ - but then the fast selection algorithm of Blum *et al.* showed this task has complexity $O(n)$ [8].

For $d = 2$ a centerpoint has depth at least $n/3$, and Jadhav and Mukhopadyay [31] showed how to compute one in $O(n)$ time by the prune-and-search paradigm. There is a lower bound of $\Omega(n \log n)$ by reduction to set equality [2] to find the depth of S but there is a tantalizing absence of any lower bound for *finding* a Tukey median. The current best algorithm for the $depth(S)$, *as well as for finding a median*, is due to Langerman and Steiger which runs in $O(n \log^3 n)$ time [36]. Also Chan [15] gave an $O(n \log n)$ time randomized algorithm using an efficient alternative to the parametric search. In $\mathbb{R}^d$, the current-best algorithms for both finding a point of depth $n/(d+1)$ and the highest-depth point take $O(n^{d-1})$ time [15].

Tukey depth of the points in $\mathbb{R}^d$ is *unimodal*: for any fixed τ, the set of points with Tukey depth at least τ is connected and convex. Unfortunately, unimodality of the Tukey depth that is implicitly used in almost all of the algorithms to search for a Tukey median isn't there for the simplicial depth anymore. Thus the best algorithm to find a simplicial median is a trivial one of computing the simplicial depth of all the points in the plane. Start by constructing an arrangement of the lines through each pair of the points. The $\binom{n}{2}$ lines cross at $\binom{\binom{n}{2}}{2}$. Euler formula in the plane implies that this arrangement has $O(n^4)$ faces or cells. It is easy to see that the simplicial depths all the points within a cell of the arrangement is the same. Once the simplicial depth of a point in a cell has been computed, the depth of all the points in a neighbor cell can be inferred in the constant time. Therefore the simplicial depth of every point in the plane and a simplicial median can be computed in $O(n^4)$ time.

Gill, Steiger, and Widgerson did give an algorithm to compute the simplicial median when the search space is limited to the given point set instead of all points in the plane [28]. Their algorithm runs in $O(n^2)$ time. They further showed that their technique can be extended to give $O(n^3)$ algorithm to compute a median among a given point set in $\mathbb{R}^3$. Bukh gave a simple proof of existence of a point with *high* simplicial depth (at least $\frac{n^3}{27}$) in the plane [13]. The proof is constructive in nature and can be implemented to compute the point of high simplicial depth in $O(n \log n)$ time using the algorithms for Ham Sandwich Cut [40].

Now given a set S of n points in $\mathbb{R}^d$, let E be the set of all $\binom{n}{d}$ $(d-1)$-simplices spanned by S. Recall that *ray-shooting depth* (called RS-depth from now on) of a point z is defined as the minimum number of simplices in E that any ray from z must intersect. While the problem of existence of a point with high RS-depth is open in $\mathbb{R}^d$, $d \geq 3$, it is shown in [26] that, given any S in $\mathbb{R}^2$, there exists a point with RS-depth at least $n^2/9$. An easy construction shows that this is optimal.

The topological proof given in [26] follows from a variant of Brouwer's fixed point theorem and is as such purely existential. Although a straightforward algorithm can be derived from it with running time $O(n^5 \log^5 n)$ by an exhaustive search. The main goal of this chapter is to present faster algorithm for computing RS-depth of a planar point set. Admittedly, one would like an algorithm for computing points of high RS-depth in higher dimensions as well. However, it is not clear how to extend the current topological machinery that we use for the $d = 2$ case to higher dimensions.

We will show that one can compute a point of RS-depth $n^2/9$ in the plane in $O(n^2 \log^2 n)$ time. We have also implemented some algorithms in a software package, and made it available in the statistical computing software package called R [52]. This is explained in Section 4.3. The majority of the contents of this chapter were published in [47].

4.2 Computing a point of ray-shooting depth at least $n^2/9$

In the main part of the chapter, we present an algorithm to compute a point of high ray-shooting depth. It that takes as input a finite set S of n points in the plane and finds a point p in $\mathbb{R}^2$ with ray-shooting depth at least $n^2/9$. Given a point $z \in \mathbb{R}^2$ and a vector $u \in \mathbb{S}^1$, denote by $\rho_{z,u}$ the closed ray emanating from z in the direction u. For any $p \in \mathbb{R}^2$, $p \neq z$, denote the closed ray emanating from z and passing through p as $\rho_{z,p}$. We will denote the unit vector in the direction of $\rho_{z,p}$ as $\delta_{z,p}$. A ray ρ is *bad* if it intersects fewer than $n^2/9$ edges (*closed segments*) spanned by the input points, and *good* otherwise. A pair of rays starting for a point in the plane define two cones. A cone is closed if it contains both of these rays. The following lemma is from [26].

Lemma 1. *For any $z \in \mathbb{R}^2$ and for any two bad rays ρ_1 and ρ_2 emanating from z, one of the two closed cones defined by ρ_1 and ρ_2 contains fewer than $n/3$ input points.*

Proof. If both cones defined by ρ_1 and ρ_2 have at least $n/3$ points, the rays ρ_1 and ρ_2 together intersect at least $2n^2/9$ edges. This is a contradiction to the assumption that both of them are bad rays. □

Using the above lemma, we prove the following.

Lemma 2. *For any $z \in \mathbb{R}^2$, there exists a cone $\mathcal{C}_z$ with apex z containing more than $2n/3$ input points, and where all the rays emanating from z and contained in $\mathcal{C}_z$ are good.*

Proof. If there are no bad rays emanating from z then we take $\mathcal{C}_z$ as the entire plane. Otherwise, let ρ be a bad ray emanating from z. Let us pick a ray ρ' emanating from z such that both the closed cones defined by ρ and ρ' contain at least $n/2$ input points. Note that ρ' is a good ray (Lemma 1).

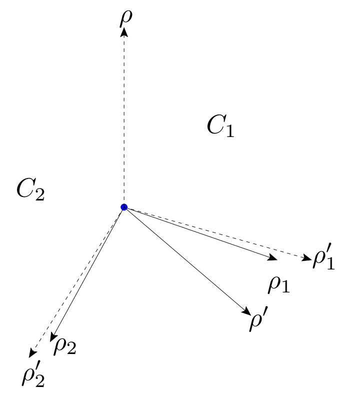

Figure 4.1: The good rays are solid, bad rays are dashed.

Imagine starting with the ray ρ', and rotating it in either direction about z as long as it is good. See Figure 4.1. Let ρ_1 (respectively ρ_2) be the last good ray before we hit a bad ray, in the counter-clockwise (resp. clockwise) direction. Clearly ρ_1 and ρ_2 pass through input points (recall that we are counting *closed* segments). Let ρ'_1 be a bad ray close-enough to ρ_1 so that there are no points of S between these rays. Similarly let ρ'_2 be a close-enough bad ray to ρ_2.

The cone C_1 defined by ρ and ρ'_1 (Figure 4.1) contains fewer than $n/3$ input points (by Lemma 1 applied to ρ and ρ'_1, as the cone complement to C_1 contains at least $n/2$ points). Similarly, the cone C_2 contains fewer than $n/3$ points. Consider now the bad rays ρ'_1 and ρ'_2 . Let C_3 be the cone defined by these rays that contains the ray ρ'. C_3 cannot contain fewer than $n/3$ points since $C_1 \cup C_2 \cup C_3$ covers $\mathbb{R}^2$ and C_1 and C_2 contain fewer than $n/3$ points each.

Hence the other cone defined by ρ'_1 and ρ'_2 contains fewer than $n/3$. Therefore, the cone C_z defined by ρ_1 and ρ_2 that contains ρ' contains more than $2n/3$ points. By construction, all rays in C_z are good. □

For any point $z \in \mathbb{R}^2$, define the good cone at z to be the closed cone C with apex z containing the maximum number of input points such that all rays emanating from z and lying in C are good. Clearly, the cone $\mathcal{C}_z$ obtained in the proof of the previous lemma is the unique good cone at the point z since it contains more than $2n/3 > n/2$ points and is maximal. For any $u \in \mathbb{S}^1$, if $\rho_{z,u} \in \mathcal{C}_z$, we say that u is a *good direction* at z. Then the cone $\mathcal{C}_z$ defines the set D_z of good directions at z. We will call the set of input points lying in $\mathcal{C}_z$ the *good set* of z and denote it by $\mathcal{G}_z$.

For computational purposes, we will need a data structure which we now describe. Let X be a set of n distinct fixed points on a circle C henceforth referred to as *locations*. Let $x_1, x_2, \cdots, x_n$ be the circular order of the points. (The precise coordinates of these points do not matter.) Let $T = \{t_1, \cdots, t_n\}$ be the set of n open interval on C defined by consecutive locations in X. Let $Y = \{Y_1, \cdots, Y_n\}$ be a set of n points, each placed at one of the locations in X. For any points $u, v \in C$, we will denote by $A_{u,v}$ the arc going from u to v counter-clockwise along C. Note that $A_{u,v}$ is different from $A_{v,u}$. For any point $q \in C$, we define its *depth* as the number of arcs $A_{u,v}$ containing q for all $u, v \in C$. Since each point in any of the open intervals t_i has the same depth, we can define the depth of t_i and the depth of any point in it. We need the data structure to support the following operations:

1. Insert an arc $A_{u,v}$, $u, v \in Y$
2. Delete an arc $A_{u,v}$, $u, v \in Y$
3. Move a point $y \in Y$ to a neighboring location.
4. Given a query arc $A_{u,v}$, $u, v \in C$ and an integer k report the first and last intervals, if any, on $A_{u,v}$ with depth smaller than k.

5. Given an integer k report an interval, if any, with depth smaller than k.

Notice that the end points of the arcs we add or delete are in Y. When we move a point $y \in Y$ to a neighboring location, the end points of the arcs incident to y move with it. However, in the fourth operation listed above, the end points of the query arc are arbitrary points in C. Building such data structures is routine in computational geometry. It is possible to build the data structure in $O(n \log n)$ time such that each of the operations take $O(\log n)$ time. We skip the easy details. Let us see how we can use this data structure to compute the good cone at any point $z \in \mathbb{R}^2$.

Lemma 3. *The good cone of any point $z \in \mathbb{R}^2$ can be computed in $O(n^2 \log n)$ time.*

Proof. We take a unit circle and fix X to be any n points $x_1, \cdots, x_n$, in that order around C. We put a point y_i at the location x_i for $i \in [1, n]$. We do an angular sorting of the input points around the point z and for every input point p_i we put a representative point y_i in the location x_r where r is the rank of p_i in the sorted order. For each pair of input points p_i and p_j, if the line $l_{i,j}$ through p_i and p_j does not pass through z, we insert either the arc A_{y_i,y_j} or A_{y_j,y_i} depending on whether z is to the left or right of $l_{i,j}$ oriented in the direction from p_i to p_j. If $l_{i,j}$ passes through z we either add both or none of the arcs depending on whether z lies on the edge p_ip_j. We then query the data structure to see if there are any intervals of depth smaller than $n^2/9$. If there are no such intervals, all rays emanating from z are good. If not, the data structure gives us a an interval I with depth less than $n^2/9$. Let z' be any point in this interval. From this, we obtain a bad ray $r_b = \rho_{z,z'}$ emanating from x. We then pick the ray r_g such that each of the closed cones defined by r_b and r_g contain at least $n/2$ points. We know that the ray r_g is good. As discussed before, we can find the good cone at z by rotating r_g to either side and stopping at the last good ray before we hit bad rays. We can find the first bad interval (interval of depth smaller than $n^2/9$) in either direction by using queries of type 4. Since we insert $O(n^2)$ arcs and each insertion takes $O(\log n)$ time, the time taken for inserting the arcs is $O(n^2 \log n)$. Once we have inserted all the arcs, it takes only a constant number of queries, each taking $O(\log n)$ time, to determine the good cone. The overall running time is therefore $O(n^2 \log n)$. $\square$

This is not the best algorithm for computing the good cone at a point. The good cone can be computed in $O(n \log n)$ time as below. But we will not need it for this paper since this is not the bottleneck for the main algorithm in this paper.

We observe that

Lemma 4. *The good cone of any point $z \in \mathbb{R}^2$ can be computed in $O(n \log n)$ time.*

We will prove a slightly more general statement from which Lemma 4 follows easily. In the following $E(\rho)$ denotes the set of edges that intersect a ray ρ, and for a pair of points a and b on a circle, $\overrightarrow{ab}$ denotes the arc from a to b in the clockwise direction. Recall that $\overrightarrow{ab} \neq \overrightarrow{ba}$.

Fact 1. *For any point z in the plane, number of edges intersecting ρ_{z,p_i} for all $p_i \in S$ can be computed in $O(n \log n)$ time.*

Proof. We describe a simple algorithm to compute the number of edges meeting ρ_{z,p_i} for all $p_i \in S$ and having the stated time complexity.

Translate the points in S so that z becomes the origin. Further assume that after sorting the points of S radially around $z = (0,0)$ $p_1, p_2, \ldots, p_n$ denotes the ordered list of points, increasing in the counter clockwise direction with p_1 lying on the positive horizontal axis. For a fixed point p_i above the horizontal axis, the line through p_i and $-p_i$[1] determines which edges, incident on p_i, intersect ρ_{z,p_1}. Explicitly, an edge $\overline{p_i p_k}$ meets the ray ρ_{z,p_1} if and only if p_k lies to the right of line through p_i and $-p_i$ and below the horizontal axis. If x_i counts the number of points in S that lie on the arc as we go from p_i to $-p_i$ in the clockwise direction, number of

[1] For a point $a = (x, y)$, $-a$ is the antipodal point of a i.e $-a = (-x, -y)$.

such edges is given by $x_i - i + 1$. We have

$$|E(\rho_{z,p_1})| = \sum_{i=1}^{k} x_i - i + 1,$$

where $p_1, \ldots, p_k$ are the points that lie on or above the horizontal axis. We use following relation to compute x_i's.

$$x_{i+1} = x_i + 1 - y_i,$$

where y_i is the number of points of $-S = \{-p_1, -p_2, \ldots, -p_n\}$ that lie on the arc $\overrightarrow{p_{i+1}p_i}$. It is easy to see that once we have radially sorted points $p_1, p_2, \ldots, p_n$ the quantities y_i and x_i can be computed in $O(n)$ time for all $1 \leq i \leq n$. We have the following relation between the number of edges meeting the rays defined by a pair of consecutive points p_i, p_{i+1} on the unit circle.

$$|E(\rho_{z,p_{i+1}})| = |E(\rho_{z,p_i})| - x_i + n - x_{i+1}.$$

This is because edges of the form p_ip_k, where $p_k \in \overrightarrow{p_i{-}p_i}$, meet ρ_{z,p_i} but do not meet $\rho_{z,p_{i+1}}$ and edges of the form $p_{i+1}p_\ell$, where $p_\ell \in \overrightarrow{-p_{i+1}p_{i+1}}$, do not meet ρ_{z,p_i} but do meet $\rho_{z,p_{i+1}}$. Edges that are not incident on either p_i or p_{i+1} must intersect both rays.

The time complexity of the algorithm is dominated by the radial sort. □

Let $\mathcal{L}$ denote the set of the $\binom{n}{2}$ lines passing through the distinct pairs of input points and let $\mathcal{A}$ be the arrangement of the lines in $\mathcal{L}$.

Lemma 5. *If $z, r \in \mathbb{R}^2$ lie in the interior of the same cell of $\mathcal{A}$ then $\mathcal{G}_z = \mathcal{G}_r$. If z lies in the interior of a cell and r lies on the boundary of the same cell, $\mathcal{G}_z \subseteq \mathcal{G}_r$.*

Proof. Let z and r be points in the interior of the same cell of $\mathcal{A}$. The angular order of the input points is the same around both points. Let $p_1, \cdots, p_k$ be the points in the good set of z in the angular order. Any ray in the good cone of z intersects some edge p_ip_{i+1} formed by two consecutive points in the good set of z. We will show that for each such edge, any ray emanating from r and intersecting that edge is also good. This will show that all points in the

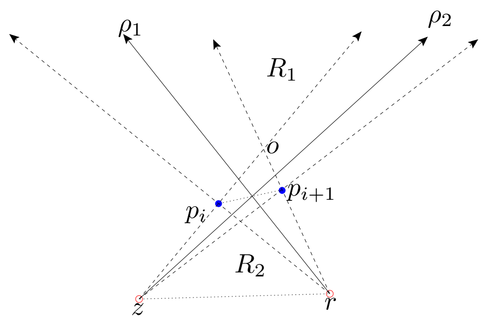

Figure 4.2: The good set is the same for all points in a cell.

good set of z are also in the good set of r i.e. $\mathcal{G}_z \subseteq \mathcal{G}_r$. The same argument with the roles of z and r exchanged shows that $\mathcal{G}_r \subseteq G_z$, implying that $\mathcal{G}_z = \mathcal{G}_r$.

Let ρ_1 be a ray emanating from r and intersecting the edge p_ip_{i+1}. Let ρ_2 be a ray emanating from z and intersecting the edge p_ip_{i+1} at the same point as ρ_1. Since p_i and p_{i+1} are in the good set of z, we know that the ρ_2 is good. We will show that ρ_1 intersects all the edges that ρ_2 intersects. Let C_1 be the cone defined by ρ_1 and ρ_2 containing the region R_1 and let C_2 be the cone defined by them containing the region R_2 (see Figure 4.2). If there is an edge that intersects, ρ_2 but not ρ_1 there must be an input point in at least one of these cones. Let A_z (respectively A_r) be the cone with apex z (resp. r), containing p_ip_{i+1} and bounded by rays through p_i and p_{i+1}. Since p_i and p_{i+1} are consecutive points in the angular order around the points z and r, there are no input points in the cones A_z and A_r. Therefore, if there are any input points in the cones C_1 or C_2, they must lie in the regions R_1 or R_2. However if there is any input point p in one of these regions then pp_i intersects the segment zr contradicting the assumption that z and r lie in the interior of the same cell. The same argument goes through if z lies in the interior of a cell and r lies on the boundary of a cell, except if r lies on the line through p_i and p_j. In this case, one can still show that $\mathcal{G}_z \subseteq \mathcal{G}_r$ but ρ_1 may intersect edges incident to p_i or p_{i+1}. $\square$

In the following, to any continuous non-self-intersecting curve ω with distinct endpoints a and b, we associate a continuous bijective function $\hat{\omega} : [0,1] \mapsto \omega$ so that $\hat{\omega}(0) = a$ and $\hat{\omega}(1) = b$. Similarly to any continuous non-self-intersecting loop ω, we associate a continuous bijective function $\hat{\omega} : \mathbb{S}^1 \mapsto \omega$. We orient $[0, 1]$ from 0 to 1 and $\mathbb{S}^1$ in the clockwise direction. This gives an orientation to ω.

Let $J \subseteq \mathbb{R}^2 \times \mathbb{S}^1$ be the set $\{(w, u) : w \in \mathbb{R}^2, u \in D_w\}$ (recall that for a point z, D_z denotes the set of good directions at z). Let π_1 and π_2 be projection functions that map a point (w, u) in J to w and u respectively. Let us also denote by ω_i, $i \in \{1, 2\}$, the curve defined by the function $\hat{\omega}_i(t) = \pi_i(\hat{\omega}(t))$. The domain of $\hat{\omega}_i$ is the same as the domain of $\hat{\omega}$ (either $\mathbb{S}^1$ or $[0, 1]$ depending on whether ω is a closed loop). The orientation of ω gives the orientation of ω_i. When ω is a closed loop, we define the winding number of ω as the winding number of $\hat{\omega}_2$.

Let γ be a non-self-intersecting continuous curve in the plane with distinct endpoints and let ω be a continuous curve in J so that $\gamma = \omega_1$. We call ω a *walk* along γ. The next lemma shows that there exists a walk along any line segment in the plane.

Lemma 6. *Let $s = (w_1, u_1)$ and $t = (w_2, u_2)$ be two points in J. Let σ be the segment joining w_1 and w_2. There exists a curve ω in J joining s and t so that $\omega_1 = \sigma$, i.e., ω is a walk over σ with endpoints s and t. Furthermore, ω can be computed in $O(n^2 \log n)$ time.*

Intuitive meaning: The statement of the lemma uses a lot of notation in order to be precise. However, since this may make it seem more complicated, here is the intuitive meaning. We want to move from w_1 to w_2 along the segment σ always maintaining a ray in the good cone of the current point. We start with the ray in the direction u_1 when we are at w_1 and the direction of the ray changes continuously as we move to w_2 and we finish with the direction u_2 at w_2. Along the motion from w_1 to w_2, we are allowed to *stand* at a point p on σ and move the ray continuously within the good cone of p.

Proof. We first compute the intersection of σ with each line in $\mathcal{L}$. Let $p_1, p_2, \cdots, p_{k-1}$ be these points in the order of intersection. Let $p_0 = w_1$ and $p_k = w_2$. Let I_i be the interval $[p_i, p_{i+1}]$. We will traverse σ from w_1 to w_2, and construct ω as we go along. The events in this traversal will be the intervals I_i and the points p_i in the order that they appear in the segment from w_1 to w_2. As we sweep, we will maintain a data structure that gives us information about the good set of the interval or point that we are currently in. Using this information, we will compute a curve along each of the points p_i and each of the intervals I_i which, when put together, gives us a walk along σ with endpoints w_1 and w_2. We construct $2k$ points in J, $(p_1 = w_1, s_1), (p_1, t_1), \ldots, (p_i, s_i), (p_i, t_i), \ldots, (p_k = w_2, t_k)$; the curve they define gives the required walk.

Each of the intervals I_i lies entirely within a single cell of $\mathcal{A}$. Therefore, there is some $g_i \in S$ so that $g_i \in \mathcal{G}_x$ for all $x \in I_i$. For point $i \in [1, \ldots, k-1]$, both g_{i-1} and g_i are in $\mathcal{G}_{p_i}$. Set $s_i = \delta_{p_i, g_{i-1}}$ and $t_i = \delta_{p_i, g_i}$; note that both these directions are in $\mathcal{D}_{p_i}$. Therefore, there is an interval $K_i \subseteq \mathcal{D}_{p_i}$ with endpoints s_i and t_i. We define K_0 as the interval contained in $\mathcal{D}_{p_0}$ with endpoints u_1 and δ_{p_0, g_0} and we define K_k as the interval contained in $\mathcal{D}_{p_k}$ with endpoints $\delta_{p_k, g_{k-1}}$ and u_2. For the interval I_i, we define the walk $\omega_{I_i} = \{(x, \delta_{x, g_i}) : x \in I_i\}$ and for each point p_i we define the walk $\omega_{p_i} = \{(p_i, x) : x \in K_i\}$. The walk ω_{p_0} starts at the point (p_0, u_1) and ends at the point (p_0, δ_{p_0, g_0}) which is where ω_{I_0} starts. ω_{I_0} ends at (p_1, δ_{p_1, g_0}) which is where ω_{p_1} starts and so on. Putting together these walks we get the required walk ω from (w_1, u_1) to (w_2, u_2).

In order to compute these walks, we will need to know the good cones in each of the intervals I_i and at each of the points p_i. We start by computing the good cone of p_0 and the good cone of some point z in I_0. The good cone at z gives us g_0 with which we compute ω_{p_0} and ω_{I_0}. We will update the data structure used to compute the good cone of z and obtain the good cone of p_1. In the data structure we have a representative y_i for each input point in p_i whose locations reflect their angular ordering around z. For convenience we will refer to the representatives by the points themselves. So, when we write "move p_i to a neighboring location", we mean "move y_i to a neighboring location". The point p_1 is the intersection of σ with some line in $\mathcal{L}$ passing through two input points a and b. There are two cases to consider depending on whether p_1 lies on the edge ab or not. Assume that p_1 lies on the edge ab. In this case we add the arc joining a and b which is not already in the data structure so that both arcs are present when we are at p_1. This reflects the fact that any ray emanating from p_1 intersects the edge ab. When we move from p_1 to I_1, we will remove the arc that was added first and keep the second one. Effectively as we move across p_1, we switch from one arc formed by a and b to the other arc formed by a and b. Assume now that p_1 does not lie on the edge ab. In this case, we move the point a to the location of b as we move from I_0 to p_1. When we move from p_1 to I_1, we move b to the previous location of a. This reflects the fact that as we move across p_1 from I_0 to I_1, the points a and b switch their positions in the angular order. When we are at p_1, they are at the same position. The rest of the sweep is done in a similar fashion. Each update takes $O(\log n)$ time. Hence the total time required for the sweep is $O(n^2 \log n)$. □

Let ω be a walk along a rectangle R in the plane and $p_1 = (w_1, u_1)$ and $p_2 = (w_2, u_2)$ be two points on ω such that w_1 and w_2 lie on different edges of R. Let σ be a chord of R joining w_1 and w_2. From Lemma 6, we obtain walk $\tilde{\sigma}$ joining p_1 and p_2 in J such that $\tilde{\sigma}_1 = \sigma$. The points p_1 and p_2 split ω into two arcs, one oriented from p_1 to p_2 and the other oriented from p_2 to p_1. The curve $\tilde{\sigma}$ splits the loop ω into two loops α and β. The loop α traverses the arc of ω from p_1 to p_2 followed by the curve $\tilde{\sigma}$ from p_2 to p_1. The loop β traverses curve σ from p_1 to p_2 followed by the the arc of ω from p_2 to p_1. Observe that the winding numbers of the loops α and β add up to the winding number of the loop ω because if we traverse α followed by β, we traverse $\tilde{\sigma}$ consecutively in opposite direction canceling its effect with respect to the winding number. Therefore, if the winding number of ω is non-zero, the winding number of one of the loops α or β is non-zero. This gives us a way to find, from any loop of non-zero winding number, a *smaller* loop of non-zero winding number.

Lemma 7. *Let R be a finite volume rectangle with a walk ω of non-zero winding number over it. There is point p inside R with ray-shooting depth at least $n^2/9$.*

Proof. Let σ be a vertical or horizontal chord of R that splits R along its longer side, into two rectangles R_1 and R_2 of equal area. From the above discussion, it follows that there is a walk of non-zero winding number along one of the rectangles R_1 or R_2. We repeat this process with that rectangle. In this process, we get nested rectangles with smaller and smaller longer side and hence converge to a point p which has a non-zero winding number over it. This means that any ray emanating from p is good and hence p has ray-shooting depth at least $n^2/9$. □

The above lemma remains true even if R is any closed curve instead of a rectangle. We state it in terms of a rectangle because that is what we use for computational purposes.

Summary of the Algorithm

Let R be a rectangle containing S. We will show that there is a walk ω along R with a non-zero winding number. We will then split R into two rectangles R_1 and R_2 using a vertical chord σ of R which bisects the set of vertices of $\mathcal{A}$ within R. We will find a walk $\tilde{\sigma}$ along σ that splits ω into two walks α and β along R_1 and R_2 respectively. One of these will have a non-zero winding number. We will replace R by the rectangle that has a walk of non-zero winding number along it and repeat the process. In each iteration, we reduce the number of vertices of $\mathcal{A}$ in R by a factor of two. Since there are at most $O(n^4)$ vertices to begin with, in $O(\log n)$ iterations, we will find a rectangle R so that there is walk ω along R with non-zero winding number and R has at most one vertex of $\mathcal{A}$. From Lemma 7, we can conclude that there is a point p of ray-shooting depth at least $n^2/9$ in the region bounded by R. Since all points in the region bounded by R belong to a cell intersecting R, we can just check the $O(n^2)$ cells intersecting R to find the required point. We will finally show that each iteration can be implemented in $O(n^2 \log n)$ time. The overall running time of the algorithm will therefore be $O(n^2 \log^2 n)$.

We now show that there is a walk of non-zero winding number along any non-self-intersecting continuous loop γ enclosing the input point set. Let $\mu = (3 - \sqrt{5})/6$ so that $\mu(1-\mu) = 1/9$. It is easy to see that for any point p on γ, a ray ρ emanating from p is good if and only if it has at least μn input points on either side of the line containing ρ. This means that any point $z \in \mathbb{R}^2$ with Tukey depth more than μn (w.r.t. S) is in the good cone of every point p on γ. Since $\mu < 1/3$, the Centerpoint theorem guarantees that there is such a point o. For any point p on γ, let $u_p \in \mathbb{S}^1$ be the direction $\delta_{p,o}$. Consider the walk ω along γ such that $\hat{\omega}(t) = (\hat{\gamma}(t), u_{\hat{\gamma}(t)})$. Clearly the winding number of ω is either $+1$ or -1 depending on whether γ itself has a winding number $+1$ or -1 around o.

For the purpose of our algorithm, we will start with a rectangle R which encloses the input points and none of the lines in $\mathcal{L}$ intersect the horizontal sides of $\mathcal{A}$. We can assume without loss of generality that no two of the input points lie on the same vertical line. Therefore, such a rectangle always exists and can be computed in $O(n^2)$ time. We then compute a point of Tukey depth at least μn. This can be done in $O(n)$ time [31]. This gives us a walk ω over R.

We can compute the number of vertices of $\mathcal{A}$ between any two vertical lines l_1 and l_2 by comparing the top to bottom order of the intersection of the lines in $\mathcal{L}$ with these two lines. Finding a vertical chord σ of R that splits the number of vertices evenly is therefore a simple slope selection problem and can be done in $O(n \log n)$ time [17]. Let a and b be the endpoints on σ and let R_1 and R_2 be the rectangles that σ splits R into.

Using Lemma 6, we compute a walk $\tilde{\sigma}$ over σ joining some points (a, u_a) and (b, u_b) on ω. $\tilde{\sigma}$ splits ω into two walks α and β along R_1 and R_2 respectively. Each of the walks consists of $O(n^2)$ pieces which form $\tilde{\sigma}$ and one piece along ω. The curve α_2 (and similarly β_2) is monotonic on each of these pieces. Hence the winding number of the walks α and β can easily be computed from these pieces. We pick one of the rectangles R_1 or R_2 which has a walk of non-zero winding number along it and recurse. We do this until we have a rectangle that contains at most one vertex of $\mathcal{A}$. From Lemma 7, we know that there is a point of ray-shooting depth at least $n^2/9$ inside the rectangle. Since the rectangle contains at most one vertex there is no cell contained completely in the interior of R. In our divide and conquer process, we have already checked all the cells crossed by R and computed a good cone for a point in it. One of those points must have ray-shooting depth at least $n^2/9$. We have therefore shown that:

Theorem 1. *Given a set S of n points in the plane, a point of ray-shooting depth at least $n^2/9$ with respect to S can be computed in $O(n^2 \log^2 n)$ time.*

4.3 Implementation

We have implemented some of the algorithms presented here into a software package. The software package includes a program to compute the ray-shooting depth of a query point, and a program to find the ray-shooting median of a given point set, an approximate median, as well as some visual data representation tools in the two dimensions. The computation of an approximate RS-median is based on the following theorem by Mustafa *et al.*

Theorem 2. *[47] Given a set P of n points in the plane, a point z of ray-shooting depth at least $(1-\epsilon)d$, where d is the maximum ray-shooting depth of any point with respect to P, can be computed in $O(1/\epsilon^8 \log^5 1/\epsilon + n)$ time.*

This has been integrated as a package in the popular statistical computing software R [52], and is available on the R public repository here:
`http://cran.r-project.org/web/packages/rsdepth/`.
One of the tools called *rsrings* gives a visual description of the contours of the points and is perfect for a study of bivariate *spacing*. For a description on theory and applications of spacings in statistics see [51]. In Figure 4.3 we use it on three sets of 500 points drawn from bivariate uniform, normal and exponential distributions at random. Observe that in all cases the rings drawn provide a near optimal estimate of the underlying distribution. Also provided in the software is *rsplot*, another graphical tool. Rsplot can be used to picture the properties of data including the location, outliers, skewness etc independent of affine transformation in the plane. The graphs constructed using rsplot include

- Median bag: This is the convex hull of the median points, the points with the highest ray-shooting depth in the plane. In the figure 4.4 this is represented as the small dark blue convex polygon containing a red point.

- RS-median point: This is the red point in the figure 4.4 - a point of the highest ray-shooting depth. If there are more than one points of highest depth, the centroid is chosen to represent this point as a convention.

- Half bag: a convex polygon that contains 50% of the data points with the highest RS-depth, i.e., $\eta(z) \geq \eta(z')$ for all z in Half bag and all z' not in Half bag.

- Fence: A convex polygon that identifies outliers in the data. Outliers are the points that so *far away* from the rest of the points that they are considered as corrupt data resulted from the errors in the measurements. The precise definition of an outlier is dependent on the individual applications and can be adjusted in the software.

Similar plots have appeared before in the statistics literature - for example *Bagplot* provides a visual depiction of data as rsrings and rsplot but uses Tukey depth is the underlying tool [27]. We note that relative to other similar software in statistics development environments, our implementation is quite efficient for practical purposes, for example, it took less than 3 seconds to compute and draw 100 rings on a set of 500 points on 2GHz Intel processor. A Bagplot implementation that uses Tukey depth to draw contours, took more than a minute on the same set of points on same machine [27]. Unfortunately we could not find any reliable software for drawing the contours based on simplicial depth for large data.

In Figure 4.4 we show an application of rsplot on the plasma readings of 60 patients. The plot on the left is an approximate one, where the Half bag is constructed by the convex hull of the $n/2$ deepest datapoints with respect to their RS-depth.

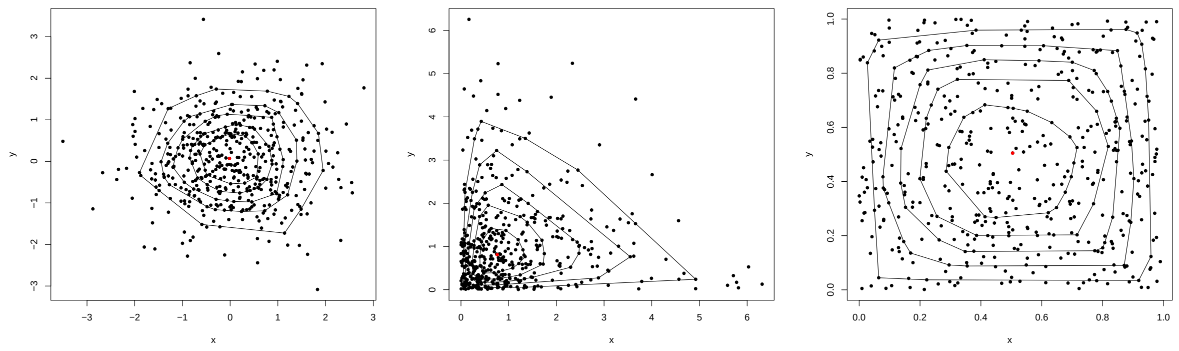

Figure 4.3: Five rings of RS-depth for sets of 500 points of random bivariate data. For the left-most figure the points were drawn from standard normal distribution with mean 0 and standard deviation 1. The points from the exponential distribution with mean 1 feature in the middle figure. And the datapoints in the rightmost figure are drawn uniformly from the unit square.

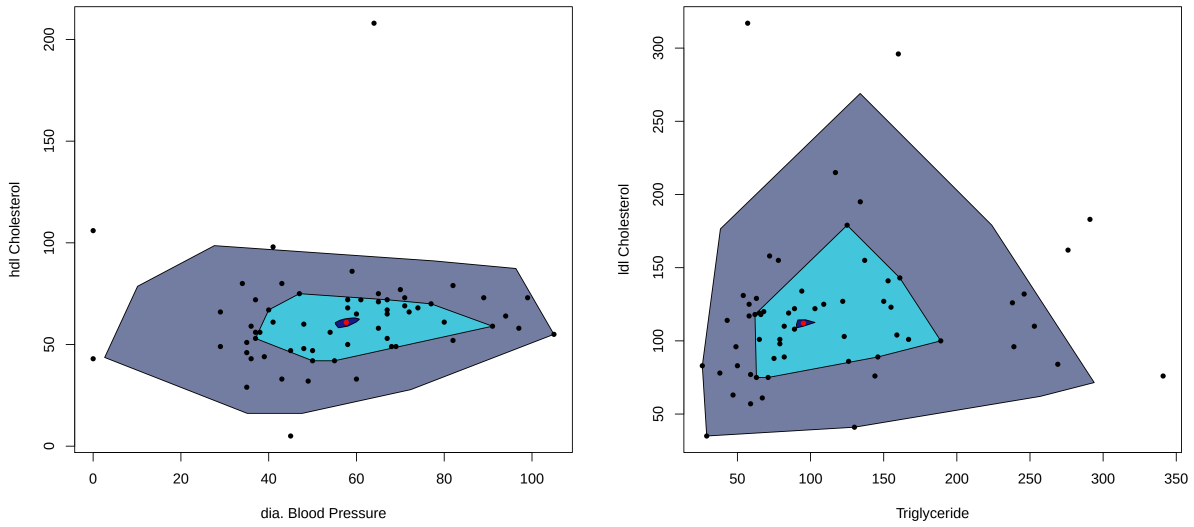

Figure 4.4: The plot on the left is rsplot of Triglyceride, and ldl Cholesterol readings of 60 patients showing the median, the region around median, the Half bag, and the fence. On the right we have a similar rsplot for diabolic BP and hdl Cholesterol readings of the patients. While studying effects of a drug, it is helpful to classify the subjects depending on how much their numbers conform to the normal or the expected behavior.

Chapter 5

Decontaminating Network

5.1 Introduction

A network is a collection of nodes and links. Goals of a robust network system include distribution and management of tasks and resources among nodes along with reliable and efficient transport of objects and information through the links. Network systems with complex structures and behaviors are ubiquitous for example neural networks, social networks, and telecommunication networks. Characteristic of a network to be able to efficiently propagate a subject, for example information, also poses as the greatest threat to the functioning of a network system; that is a network also serves as a mechanism to transport and propagate things that a system would rather want to contain. For example a network may want to detect and isolate errors, viruses, misinformation or rumors once they are introduced in the system.

Faults and viruses often spread in networked environments by propagating from site to neighboring site through the links in the network. The process is called *network contamination*. Once contaminated, a network node might behave incorrectly, and it could cause its neighboring nodes to become contaminated as well, thus propagating faulty computations. The propagation patterns of faults can follow different dynamics, depending on the behavior of the affected node, and topology of the network. At one extreme we have a full spread behavior: when a site is affected by a virus or any other malfunction, the malfunction can propagate to all its neighbors. Other times, faults propagate only to sites that are susceptible to be affected. The definition of susceptibility depends on the application but often it is based on local conditions. For example, a node could be vulnerable to contamination if a majority of its neighbors are faulty, and immune otherwise (e.g., see [34], [35], [41]); or it could be immune to contamination for a certain amount of time after being repaired (e.g., see [20], [25]).

In this chapter we consider a propagation of faults based on what we call *temporal immunity*: a clean node can sustain to be exposed to contaminated nodes for a predefined amount of time after which it becomes *contaminated*. Actual decontamination is performed by mobile cleaning agents which move from host to host over network connections.

5.1.1 Previous Work

Graph Search

The decontamination problem considered in this paper is a variation of a problem extensively studied in the literature known as *graph search*. The graph search problem was first introduced by Breish in [12], where an approach for the problem of finding an explorer that is lost in a complicated system of dark caves is given. Parsons ([49][50]) proposed and studied the pursuit-evasion problem on graphs. Members of a team of searchers traverse the edges of a graph in pursuit of a fugitive, who moves along the edges of the graph with complete knowledge of the locations of the pursuers. The efficiency of a graph search solution is based on the size of the search team. The size of smallest search team that can *clear* a graph G is called the search number, and is denoted in the literature by $s(G)$. In [45], Megiddo *et al.* approached the algorithmic question: *Given an arbitrary G, how should one calculate* $s(G)$*?* They proved that for arbitrary graphs, determining if the search number is less than or equal to an integer k is NP-Hard. They also gave algorithms to compute $s(G)$ where G is a special case of trees. For their results, they used the fact that recontamination of a cleared vertex does not help reduce $s(G)$, which was proved by LaPaugh in [37]. A search plan for G that does not involve recontamination of cleared vertices is referred to as a *monotone* plan.

Decontamination

The model for decontamination studied in literature is defined as follows. A team of agents is initially located at the same node, the homebase, and all the other nodes are contaminated. A decontamination strategy consists of a sequence of movements of the agents along the edges of the network. At any point in time each node of the network can be in one of three possible states: clean, contaminated, or guarded. A node is guarded when it contains at least one agent. A node is clean when an agent passes through it and all its neighboring nodes are clean or guarded, contaminated otherwise. The solution to the problem is given by devising a strategy for the agents to move in the network in such a way that at the end all the nodes are clean.

The tree was the first topology to be investigated. In [6], Barrière *et al.* showed that for a given tree T, the minimum number of agents needed to decontaminate T depends on the location of the homebase. They gave the first strategies to decontaminate trees.

In [24], Flocchini *et al.* consider the problem of decontaminating a mesh graph. They present some lower bounds on the number of agents, number of moves, and time required to decontaminate a $p \times q$ mesh $(p \leq q)$. They showed that at least p agents, pq moves, and $p + q - 2$ time units are required to solve the decontamination problem. Decontamination in graphs with *temporal immunity*, which is similar to the model of decontamination used in this paper, was first introduced in [25] where the minimum team size necessary to disinfect a given tree with temporal immunity τ was derived. The main difference between the classical decontamination model, and the new model in [25] is that once an agent departs the decontaminated node is immune for a certain $\tau \geq 0$ (where $\tau = 0$ corresponds to the classical model studied in the previous work) time units to viral attacks from infected neighbors. After the temporal immunity time τ has elapsed, recontamination can occur.

Some further work in the same model was done in [21], where a two dimensional lattice is considered.

5.1.2 Definitions and Terminology

We will only deal with connected finite graphs without loops or multiple edges. For a graph $G = (V, E)$, and a vertex $v \in V$ let $N(v)$, the *neighborhood* of v, be the set of all vertices w such that v is connected to w by an edge. Let $deg(v)$ denote the *degree* of a vertex v which is defined to be the size of its neighborhood. The maximum and minimum degrees of any vertex in G are denoted by $\Delta(G)$ and $\delta(G)$ respectively. The *shortest distance* between any two vertices $u, v \in V$ is denoted by $dist(u, v)$ and the *eccentricity* of $v \in V$ is the maximum $dist(u, v)$ for any other vertex u in G. The radius of a graph, $rad(G)$, is the minimum eccentricity of any vertex of G and the vertices whose eccentricity is equal to $rad(G)$ are called the *center vertices*. The diameter of a graph, $diam(G)$, is the maximum eccentricity over all the vertices in G.

K_n is the complete graph on n vertices. $K_{m,n}$ denotes the complete bipartite graph where the size of two partitions is m and n. An acyclic graph is known as a tree and a vertex of degree 1 in a tree is known as a *leaf* of the tree. The rest of the tree terminology used is standard. A star graph, S_n, is a tree on $n + 1$ vertices where one vertex has degree n and the rest of the vertices are leaves. Sometimes a single vertex of a tree is labeled as the *root* of the tree. In this case the tree is known as a *rooted* tree. If we remove the root vertex from a rooted tree it decomposes into one or more subtrees; each such subtree along with the root is called a *branch*, denoted by B_i, of original tree. Similarly, an *arm* is the set of vertices that lie on the path from root to a leaf, denoted by A_i.

Other classes of graphs will be defined as and when needed.

5.1.3 Decontamination Model Specification

Our decontamination model is a synchronous system. We assume that initially, at time $t = 0$, all vertices in the graph are contaminated. A *decontaminating agent* (henceforth referred to as an agent) is an entity, or a marker, that can be placed on any vertex. A concept similar to this is referred to in the literature as a *pebble* [16]. Assume that at some time step k, an agent is at $v \in V$. Then at the next time step, we may move the agent to any of the neighbors of v. Vertices visited in this process are marked *decontaminated*, or *disinfected*. Any vertex that the agent is currently placed on is considered to be decontaminated.

A decontaminated vertex can get contaminated by uninterrupted exposure, for a certain amount of time, to a contaminated vertex in its neighborhood. For decontaminated v if there is no agent placed on v but some neighbor of v is contaminated, we say that v is *exposed*. For a decontaminated vertex v we define the *exposure time* of v, $\Xi(v)$, as the duration of time v has been exposed. Every time an agent visits v, or all vertices in $N(v)$ are decontaminated, we reset $\Xi(v) = 0$. We say that G has temporal immunity $\tau(G)$ if a decontaminated vertex $v \in V$ can only be recontaminated if for uninterrupted $\tau(G)$ time units, there is a neighbor of v (not necessarily unique) that is contaminated and an agent does not visit v during that time period. Note that for any decontaminated vertex v we have that $0 \leq \Xi(v) \leq \tau(G) - 1$. Given a graph

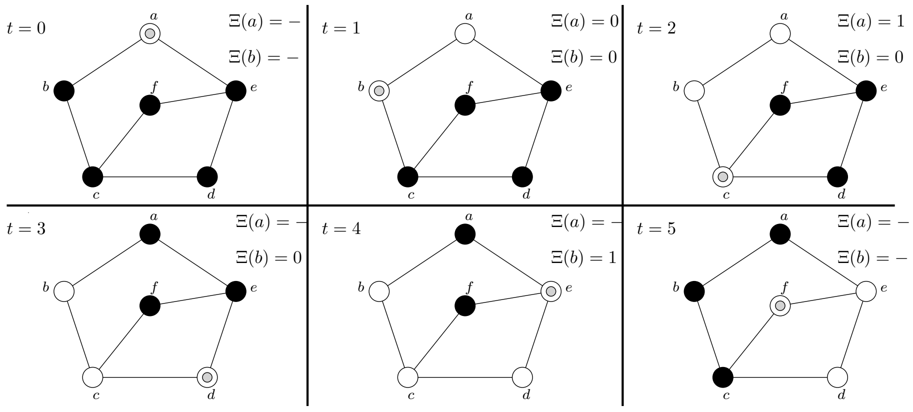

Figure 5.1: Figure illustrates variation in exposure times of vertices a and b at different time steps as the agent tries to decontaminate G, with $\tau(G) = 2$.

G, a temporal immunity τ and n agents, our goal is to devise a decontamination strategy, which consists of choosing an initial placement for the agents and their movement pattern so that we can reach a state where all the vertices of G are simultaneously decontaminated and we call the graph fully decontaminated. A strategy is called *monotone* if a decontaminated vertex is

Graph Topology	**Upper Bound on ι**	**Lower Bound on ι**
Path P_n	0 [Proposition 1]	0 [Proposition 1]
Cycle C_n	2 [Proposition 2]	2 [Proposition 2]
Complete Graph K_n	$n-1$ [Theorem 1]	$n-1$ [Theorem 1]
Complete Bipartite Graph $K_{m,n}$, with $m \leq n$	$2(m-1)$[Theorem 2]	$2(m-1)$[Theorem 2]
Spider Graph on $n+1$ vertices	$4\sqrt{n}$ [Corollary 1]	-
Tree on n vertices	$30\sqrt{n}$ [Theorem 6]	-
Mesh $m \times n$	m [Theorem 4]	$\frac{m}{2}$ [Theorem 5]
Planar Graph on n vertices	$n-1$ [Theorem 8]	$\Omega(\sqrt{n})$ [Corollary 4]
General Graphs	$n-1$ [Theorem 8]	$n-1$ [Theorem 1]

Table 5.1: A summary of our results.

never recontaminated and is called *nonmonotone* otherwise. The *Immunity number* of G with k agents, $\iota_k(G)$, is the least τ for which full decontamination of G is possible. It is trivial to see that $\iota_k(G)$ is always finite for $k \geq 1$. In particular for a connected graph G on n vertices, $\iota_k(G) \leq 2(n-1)$ for $k \geq 1$ as the depth first traversal of the spanning tree of G takes exactly $2(n-1)$ steps. However, in this paper we focus on the decontamination of graph by a single agent; this gives us the liberty to use shortened notation $\iota(G)$, and just ι when the graph is obvious from context, to mean $\iota_1(G)$, the immunity number of a graph using a single agent. In section 5.2 we prove bounds on ι for some simple graphs. In section 5.3 we give asymptotically sharp upper and lower bounds on $\iota(G)$ where G is a mesh graph. We also give algorithms to decontaminate several graph topologies. Results are outline in the table below.

5.2 Some Simple Graphs

We begin with the simple case when the graph that we want to decontaminate is a path.

Proposition 1. *Let P_n be a path on n vertices, then $\iota(P_n) = 0$, for all $n \geq 1$.*

It is easy to see that we do not need any temporal immunity to decontaminate the entire path if we start with our agent at one leaf vertex and at each time step we move it towards the other end until we reach it at $t = n-1$.

A cycle can be decontaminated using a similar strategy.

Proposition 2. *If C_n is a cycle on n vertices, $\iota(C_n) = 2$, for all $n \geq 4$.*

Proof. To see that $\iota(C_n) \leq 2$ set the temporal immunity $\tau = 2$ and begin with the agent at any vertex of the cycle. At $t = 1$ choose one of it neighbors to move to. Henceforth, for $t = k \geq 2$, we always move our agent in a fixed, say clockwise, direction. It is straightforward to verify that we will end up with a fully decontaminated graph in at most $2n$ time steps. Note that this is a nonmonotone strategy.

If we set temporal immunity $\tau = 1$ then we will show that we can never decontaminate more than two (adjacent) vertices of the cycle. Suppose that four vertices v_n, v_1, v_2, v_3 appear in the cycle in that order. Assume that at time step $t = 0$ the agent is placed at v_1 and, without loss of generality, it moves to v_2 at the next time step. At $t = 2$ if the agent moves to v_3 then v_1 becomes contaminated due to its exposure to v_n and we end up with only v_2 and v_3 decontaminated which is the same as not having made any progress. If, on the other hand, the agent had moved back to v_1 at $t = 2$ we would again have ended up with no progress since the agent would still have the same constraints on proceeding to its next vertex, therefore $\iota(C_n) > 1$. □

Remark 1. The bound presented in Proposition 2 is only tight because of our definition of τ as an integer. Otherwise we observe that there always exists a strategy to decontaminate C_n with $\tau = 1 + \epsilon$ for any real number $\epsilon > 0$ in finite time; in fact the same strategy as outlined in the proof of the upper bound above will work.

Path and cycle happen to be the simplest possible graphs that can be decontaminated easily with optimal constant immunity numbers as seen above. We now consider some dense graphs and show that they may require a much larger value of τ.

Theorem 1. *Let K_n be a complete graph on n vertices, then $\iota(K_n) = n - 1$ for all $n \geq 4$.*

Proof. Let the vertex set $V = \{v_1, v_2, \ldots, v_n\}$. Since the graph is fully connected, we can decontaminate K_n by making the agent visit all the vertices sequentially in any order giving us $\iota(K_n) \leq n - 1$.

To see that this bound is actually tight we need to show that temporal immunity of $n - 2$ is not good enough for full decontamination. For this purpose set $\tau = n - 2$ and suppose that at time step $t = k$ we have somehow managed to decontaminate all the vertices of K_n except one last vertex, say, v_n. Assume without loss of generality that the agent is at v_{n-1}. As long as the complete graph is not fully decontaminated, all the vertices which do not have the agent placed on them are exposed. This implies that the vertices $v_1, \ldots, v_{n-2}$ have all been visited by the agent in the last $n - 2$ time steps, that is, $\Xi(v_i) < n - 2$ for $1 \leq i \leq n - 2$. It also implies that since there is one agent, all these vertices have different exposure times, meaning that there is one vertex, say v_1, such that $\Xi(v_1) = n - 3$. At time step $k + 1$, if the agent moves to v_n and decontaminates it, then v_1 becomes contaminated hence we make no progress; there is still one contaminated vertex remaining in the graph. If on the other hand agent x is moved to v_1 to avoid its contamination, we will again have not made any progress. Moving the agent to any other vertex at $t = k + 1$ actually increases the number of contaminated vertices in the graph. □

The immunity number of complete bipartite graph depends upon the size of smaller partition.

Theorem 2. *Let G be a complete bipartite graph on the vertex sets A and B where $|A| = m, |B| = n$ such that $3 \leq m \leq n$, then $\iota(G) = 2m - 1$.*

Proof. Let $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_n\}$. Set the temporal immunity $\tau = 2m - 1$ and place an agent at a_1 at $t = 0$. Now we cycle through the vertices in A and B in an interleaved sequence as follows:

$$a_1, b_1, a_2, b_2, a_3, b_2, \ldots, a_m, b_m, a_1, b_{m+1}, a_2, b_{m+2}, \ldots, b_n.$$

When $t < 2m$ none of the vertices are exposed long enough to be recontaminated. At $t = 2m$ the agent returns to a_1, and thereafter none of the decontaminated vertices in B remain exposed while the vertices of A keep getting visited by the agent before their exposure time reaches τ. It follows that this monotone strategy fully decontaminates G in $2n - 1$ time steps.

Our claim is that if $\tau < 2m - 1$ then it is not possible to fully decontaminate a partition during any stage of a given decontamination strategy. Consider a strategy that aims to fully decontaminate A at some point (and B is never fully decontaminated before that). Suppose that at time $t = k$ there remains exactly one contaminated vertex in A (and that there were two contaminated vertices in A at $t = k - 1$). Note that this implies that the agent is at some vertex in A at $t = k - 1$. Since B has never fully been decontaminated, it follows that there exists a vertex $a_j \in A$ such that $\Xi(a_j) = 2m - 3$. Since it is a bipartite graph, it will take at least two additional time steps to reach the last contaminated vertex of A, and if the temporal immunity is less that $2m - 1$ the agent will fail to decontaminate A fully.

In the case where the decontamination strategy requires that B is fully decontaminated before A, similar argument gives us a lower bound of $2n - 1$ on $\iota(G)$ but we have already given a strategy that decontaminates A first which gives a better upper bound. □

5.3 Spiders, k-ary Trees, and Mesh Graphs

Two important network topologies are star and mesh. They are extreme examples of centralization and decentralization respectively. In the following we study our problem on star, *spider* (a generalization of star), k-ary trees, and mesh graphs. Some of the ideas and proof techniques developed in this section will feature again in the proof of the upper bound on immunity number for general trees that will be treated in the next section.

5.3.1 Spider and k-ary Trees

Let S be a star graph. The simple strategy of starting the agent at the center vertex and visiting each leaf in turn (via the center) gives us the best possible bound on $\iota(S_n)$.

Lemma 1. *Temporal immunity $\tau = 1$ is necessary and sufficient for any star graph.*

Proof. The strategy outlined above gives us the upper bound of $\iota(S_n) \leq 1$. The matching lower bound argument is straightforward and we omit the details. □

The *spider* is a graph that is structurally similar to a star graph. A spider is a tree in which one vertex, called the *root*, has degree at least 3, and all the rest of the vertices have degree at most 2. Equivalently a spider consists of k vertex disjoint paths all of which have one endpoint that is connected to the root vertex. Such a spider is said to have k *arms*.

Let S be a spider such that the degree of the root is Δ. If m is the length of the longest arm of S then using a naive monotone strategy of visiting each arm sequentially, starting at the root and traversing each arm to the end and returning to the root shows that temporal immunity $\tau = 2m$ is enough to fully decontaminate S. A better bound may be obtained if we allow nonmonotonicity. Set $\tau := m$. Give arms an arbitrary order $A_1, \dots A_\Delta$ and decontaminate arms in this order. If the agent has visited all Δ arms and there is still some contaminated vertices in the graph we repeat this process of decontaminating arms in order. It is easy to verify that eventually (after possibly multiple rounds) this process ends. However one can obtain an even better estimate on $\iota(S)$.

Theorem 3. *Let S be a spider on n vertices such that the degree of the root is Δ. If m is the length of the longest arm of S then $\iota(S) \leq \Delta + \sqrt{\Delta^2 + 4m}$.*

Proof. Arbitrarily order the arms of the spider $A_1, A_2, \dots, A_\Delta$ and let the temporal immunity $\tau = t_0$. Thus the agent, when starting from the root, can decontaminate t_0 vertices on an arm before the exposed root gets recontaminated. Our strategy is going to be an iterative one and in each iteration, we are going to let the root get contaminated just once in the beginning, and after we decontaminate it, we will make sure that it does not get recontaminated during the course of that iteration. At the end of iteration, j, we will have decontaminated all the arms of the spider from A_1 to A_j along with the root. Since this is going to be a nonmonotone strategy, parts or whole of these arms may be recontaminated during the course of the rest of the algorithm. At the first iteration we start from the root, traverse A_1 to the end and return to the root. We proceed to decontaminate the rest of the spider using the following strategy. At the beginning of jth iteration, our agent is at the root of the spider and all the arms from $A_1, \dots, A_{j-1}$ are fully decontaminated whereas $A_j, \dots, l_\Delta$ are all fully contaminated (except for the root). The agent traverses each arm of the spider up to the farthest contaminated vertex and returns to the root in sequence starting from A_j down to A_1. Recall that we will allow the root to get contaminated just once in this iteration, that is, when our agent is traversing A_j. We want to fine tune our the temporal immunity τ such that once the agent returns after visiting all the vertices in A_j, during the rest of the iteration when the agent is visiting other arms, the root never gets contaminated again.

Let t_1 be the total time needed to traverse the arms $A_j, \dots, A_2$ after the root has been recontaminated (when the agent reached vertex t_0 of A_j). Then

$$t_1 < 2m + 2(j-1) \times \frac{t_0}{2} \tag{5.1}$$

where the last term is the result of the constraint that the root may not be recontaminated in the current iteration. Now during the time t_1 at most $t_0/2$ vertices of A_1 could have been contaminated (once again to avoid recontamination of the root when we visit A_1). But that would have taken $t_0^2/2$ time units. Therefore:

$$t_1 = \frac{t_0^2}{2} \leq 2m + 2(j-1) \times \frac{t_0}{2}. \tag{5.2}$$

Solving (5.2) and using the fact that we get the worst bound at $j = \Delta$ we conclude that

$$\tau = t_0 < \Delta + \sqrt{\Delta^2 + 4m}.$$

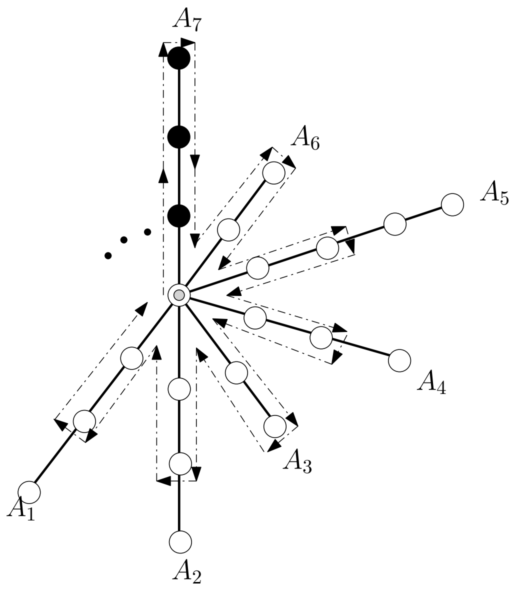

Figure 5.2: The agent is at the root. Arms $A_1, \ldots, A_6$ are decontaminated, represented as white dots. Dotted line segments show the path followed by the agent in 7th iteration to decontaminate A_7.

□

Corollary 1. *If S is a spider on n vertices then $\iota(S) = O(\sqrt{n})$.*

Proof. Let S be rooted at a vertex r. If $deg(r) = \Delta \le \sqrt{n}$, if follows from Theorem 3 that $\iota(S) \le \Delta + \sqrt{\Delta^2 + 4m} \le 4\sqrt{n}$ which gives the claim. So, without loss of generality, assume that $\Delta > \sqrt{n}$.

Let $A_1, A_2, \ldots, A_\Delta$ denote the arms of S with $|A_i| \le |A_j|$, for all $i \le j$. Again without loss of generality for some k the number of vertices in A_i is less than $\sqrt{n}$ for $i \le k$ and more than or equal to $\sqrt{n}$ for all $k < i \le m$. Now consider a modified spider $S^* = S \setminus \bigcup_{1 \le i \le k} A_i$ with r as root. By the pigeon hole principle, $\Delta(S^*) \le \sqrt{n}$. So we can apply the technique used in the proof of Theorem 3 to decontaminate S^* with $\tau \le 4\sqrt{n}$. Once S^* is decontaminated, we use the following Lemma to decontaminate $(S \setminus S^*) \cup \{r\}$.

Lemma 2. *Any k-ary tree T with height h can be decontaminated with $\tau = 2h - 1$ using a monotone algorithm.*

The bound will follow because the height of this tree is less than $\sqrt{n}$ and an already decontaminated S never gets recontaminated by the monotonicity in Lemma 2.

Proof. First label the leaf vertices of T so that $l_1, l_2, l_3, \ldots$ represents the order in which the leaves are visited if an in-order depth-first traversal is performed on T, starting from the root vertex. It is straightforward to verify that if we start with the agent at the root, and visit each leaf in order $l_1, l_2, l_3, \ldots$ returning to the root every time before visiting the next leaf, then $\tau = 2h - 1$ would be enough to decontaminate the entire k-ary tree. Also note that once decontaminated any leaf l_i is never exposed again, and all non-leaf vertices, once decontaminated, are exposed for at most $2h - 1$ time units. Monotonicity follows. □

The proof of Lemma 2 completes the proof of Corollary 1. □

A *perfect* or *full* k-ary tree is a rooted tree where every node has k children except the leaves. We observe the following corollary of Lemma 2.

Corollary 2. *Let T be a perfect k-ary tree on n vertices, then $\iota(T) = O(\log n)$.*

In case of a binary tree the bound on temporal immunity can be slightly improved if we use the above strategy to first fully decontaminate the subtree rooted at the left child of the root, and then use the same method to decontaminate the subtree rooted at the right child of the root. Thus,

Observation 1. *A binary tree with height h can be decontaminated with an temporal immunity of $2h - 3$.*

5.3.2 Decontaminating a Mesh

A $p \times q$ mesh is a graph that consists of pq vertices. It is convenient to work with planar drawing of the graph where the vertices of $G = (V, E)$ are embedded on the points with the integer coordinates of Cartesian plane. The vertices are named $v_{(i,j)}$ for $1 \leq i \leq q$, $1 \leq j \leq p$

corresponding to their coordinates in the planar embedding. There is an edge between a pair of vertices if their euclidean distance is exactly 1. We can partition V into the column sets $C_1, C_2, \ldots, C_q$ so that $C_i = \{v_{(i,j)} : 1 \leq j \leq p\}$ for all $1 \leq i \leq q$. Row sets $R_1, R_2, \ldots, R_p$ are defined analogously, i.e., $R_j = \{v_{(i,j)} : 1 \leq i \leq q\}$.

A simple approach to fully decontaminate a $p \times q$ mesh would be to place our agent at $v_{(1,1)}$ at $t = 0$, proceed to visit all vertices in the column till we reach $v_{(1,p)}$, move right one step to $v_{(2,p)}$ and proceed all the way down to $v_{(2,1)}$. This process may now be continued by moving the agent to $v_{(3,1)}$ and going on to decontaminate the entire graph column by column until we reach the last vertex. Clearly a temporal immunity of $2p - 1$ is enough for this strategy to monotonically decontaminate the entire graph. In [19] the same strategy was used, albeit under a slightly different notion of *temporal immunity*, to get a similar upper bound. However, once again we can improve this bound by resorting to a nonmonotone strategy.

Theorem 4. *Let G be a $p \times q$ mesh where $p \leq q$. Then $\iota(G) \leq p$.*

Proof. We will describe a strategy to decontaminate G in which we decontaminate each column nonmonotonically. However, once we declare a column to be decontaminated, we do not allow any of its vertices to be contaminated again.

Set the temporal immunity $\tau = p$ and start with the agent at $v_{(1,1)}$. Proceed all the way up to $v_{(1,p)}$, move the agent to the next column onto $v_{(2,p)}$, and then start traversing down the column until we reach $v_{(2,\lceil \frac{p}{2} \rceil+1)}$. Note that the vertices of C_1 had started getting recontaminated when the agent reached $v_{(2,p-1)}$ because the exposure time of $v_{(1,1)}$ became equal to τ at that point. Now move the agent back to C_1 onto $v_{(1,\lceil \frac{p}{2} \rceil+1)}$ and proceed all the way down back to $v_{(1,1)}$. We declare that C_1 has been decontaminated and none of its vertices will be recontaminated during the course of decontamination of the rest of the graph. It is pertinent to note that at this point, $\Xi(v_{(2,p)}) = \tau - 1 = p - 1$. To decontaminate the rest of the columns we use the following scheme. Assume that we have declared all the columns $C_1, C_2, \ldots, C_k$ to be decontaminated and our agent is at $v_{(k,1)}$. We also know that $\Xi(v_{(k+1,p)}) = \tau - 1$. We move the agent to the

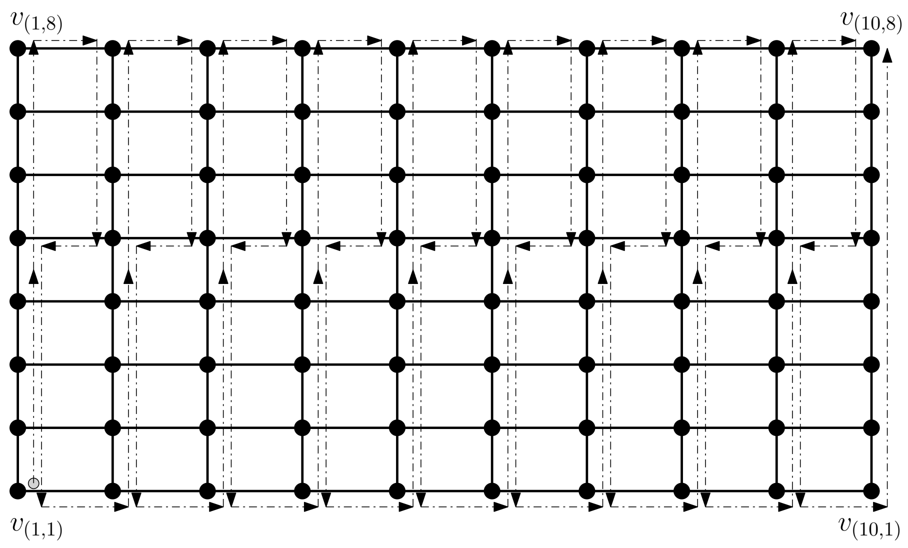

Figure 5.3: Dotted line segments outline the path agent to decontaminate mesh graph with $\tau = 8$. Once agent returns to $v_{(1,1)}$, we declare first column decontaminated, and proceed to first vertex of second column, and henceforth.

next column onto $v_{(k+1,1)}$. At this point $v_{(k+1,p)}$ becomes contaminated leaving $v_{(k,p)}$ exposed. We follow the same strategy as the one that we followed when we were decontaminating C_1. We move the agent all the way up to $v_{(k+1,p)}$, move to C_{k+2}, traverse all the way down to $v_{(k+2,\lceil \frac{p}{2} \rceil+1)}$, revert back to C_{k+1} and move back down to $v_{(k+1,1)}$ declaring column C_{k+1} to be decontaminated. None of the vertices in C_k will be recontaminated since $v_{(k,p)}$ had the maximum exposure time due to $v_{(k+1,p)}$, and we were able to decontaminate $v_{(k+1,p)}$ before $v_{(k,p)}$ got contaminated. Similarly, it is not difficult for the reader to verify that none of the rest of the vertices of C_k are exposed long enough to be recontaminated. $\square$

Corollary 3. *Let G be a mesh on n vertices, then $\iota(G) \leq \sqrt{n}$.*

Remark 2. Strategy used in proof of Theorem 4 can also be used to decontaminate a *cylinder graph* (a mesh graph with an edge between the leftmost and the rightmost vertices on each row).

In the following we present an asymptotically sharp lower bound for mesh graphs, but first we would like to establish a graph isoperimetric result that we use in proof of lower bound.

Lemma 3. *Let $G = (V, E)$ be an $\sqrt{n} \times \sqrt{n}$ mesh graph, then for any $W \subset V, |W| = \frac{n}{2}$, size of a maximum matching between W and its complement has size at least $\sqrt{n}$.*

Proof. For ease of understanding let us say that a vertex is colored white if it is in set W, and black otherwise. An edge is monochromatic if both its endpoints have the same color, and bi-chromatic otherwise. Let $R_1, R_2, \ldots, R_{\sqrt{n}}$, and $C_1, C_2, \ldots, C_{\sqrt{n}}$ be the row and column sets respectively. We observe following four possible cases:

Case 1. *For each row R_i, $0 < |R_i \cap W| < \sqrt{n}$:*

Since R_i contains vertices of both colors, it is clear that there will be at least one bi-chromatic edge. We pick one such edge from each R_i. As these edges are disjoint, we have a matching of size at least $\sqrt{n}$.

Case 2. *There exist two rows R_i, R_j, such that $|R_i \cap W| = 0$, and $|R_j \cap W| = \sqrt{n}$:*

We interchange the roles of rows and columns. The claim then follows from Case 1.

Case 3. *There exists a row R_i, such that $|R_i \cap W| = 0$, and for every row R_j, $|R_j \cap W| \neq \sqrt{n}$:*

We present below a scheme to match vertices in this case.

We will use two markers b (for bottom row), and c (for current row). In the beginning, both point to the first row of the mesh, i.e, $b := c := 1$.

1. Locate the minimum $x \geq c \geq b$, such that $R_x \cap W = \emptyset$. If we can not find such an x, go to Step 3.

   (a) Now locate the maximum $y < x \leq b$, such that $|R_y \cap W| \geq x - y + 1$. For all the white vertices in R_y we have black vertices in corresponding columns of R_x. So for each such column there exists a pair of rows R_i, R_{i+1} with a bi-chromatic edge in that column where $y \leq i < x$. We set $c := x + 1$, and call rows R_j *matched* if $y \leq j \leq x$.

   (b) If we cannot find such a y, then we look for a minimum $z > x$, such that the row corresponding to z contains at least $z - x + 1$ points from W, i.e., $|R_z \cap W| \geq z - x + 1$.

For all the white vertices in R_z, we can find bi-chromatic edges as above. We set $c := z + 1$ and $b := x + 1$ and we call rows R_j *matched* if $x \leq j \leq z$.

2. Repeat Step 1. Failure to find both y and z at any step would imply a contradiction because there are not enough black vertices as assumed. In the worst case

$$\left|\bigcup_{i=1}^{x-1} R_i\right| \leq (x-1)\frac{\sqrt{n}}{2}$$

(alternating complete black and white rows), and

$$\left|\bigcup_{j=x+1}^{\sqrt{n}} R_j\right| < (\sqrt{n} - (x+1))\frac{\sqrt{n}}{2}.$$

3. Match all unmatched rows as in Case 1.

Case 4. *There exists a row R_i, such that $|R_i \cap W| = \sqrt{n}$ and for every row R_j, $|R_j \cap W| \neq 0$:* The claim in this case follows directly from the proof of Case 3 by reversing the roles of W and $\overline{W}$.

□

This concludes the proof of Lemma.

Note that bound in Lemma 3 is tight when W is a *rectangular* subgrid. We do not know of a tight example which is not rectangular in shape. We observe that since $\Delta = 4$ for mesh, Lemma 3 also follows from vertex and edge isoperimetric inequalities proved in [9][10] up to a constant factor.

Theorem 5. *If G is a $p \times q$ mesh with $p \leq q$, then $\iota(G) > \frac{p}{2}$.*

Proof. Let us assume the opposite, i.e, that a decontaminating algorithm exists with $\tau = \frac{p}{2}$. For simplicity assume that G is a $p \times p$ mesh and ignore the agent's moves for the remaining $p \times (q-$

$p)$ vertices if any. Let $n = p^2$, then at some time step during this algorithm we will have exactly $\frac{n}{2}$ decontaminated vertices. Lemma 3 implies that at this stage at least p vertices of G are exposed through at least p disjoint edges to contaminated vertices. By considering all possible moves of the agent for next $\frac{p}{2}$ steps it is clear that at least $\frac{p}{2}$ vertices will be recontaminated, and no matter what the agent does this can decontaminate at most $\frac{p}{2}$ vertices and thus make no progress at all. It already gives that number of decontaminated vertices can never exceed $\frac{n}{2} + \frac{p}{2}$. It follows that no decontaminating algorithm exists with assumed temporal immunity.

5.4 General Trees

To upper bound ι for general trees, we will try to adapt the strategy used to decontaminate k-ary trees. The simplest approach is to apply naively the same strategy on a given tree T as before, this time considering the center vertex (choose one arbitrarily if there are two center vertices) of the tree to be the *root* and then visiting each of the leaves of T in the depth first search discovery order, every time returning to the center vertex, as in the previous case. It is clear that a temporal immunity $\tau = 2 \cdot rad(T) = diam(T) + 1$ is sufficient to fully decontaminate T but the diameter of a tree on n vertices can easily be $O(n)$. However, we can use nonmonotonicity to our advantage by letting a controlled number of vertices get recontaminated so that we get a much stronger bound even for trees with large diameters.

We will need the following lemma which describes a monotone strategy to decontaminate trees with small height.

Lemma 4. *Any rooted tree T with height h can be decontaminated with temporal immunity $\tau \geq \alpha h$, in time cn, where n is the number of vertices in T, and $c \leq 4\frac{\alpha-1}{\alpha-2}$ for any positive number $\alpha > 2$.*

Proof. Assuming an arbitrary tree with height and temporal immunity τ as above, we present an algorithm with claimed time complexity.

Define *level* of a node as the distance between the root and the node. Without loss of generality we choose a node r to be the root such that every vertex in the tree is at level less than h. Let l be the maximum integer such that there exists a subtree P_1 rooted at p_1 with $|P_1| > h\frac{\alpha}{2} - h$ at level l, and let $s_1, s_2, \ldots, s_m$ be the children of one such subtree. For all i let S_i be subtrees rooted at s_i, then $|S_i| < h\frac{\alpha}{2} - h$ by maximality of l. Now let j be the largest integer such that $|S_1 \cup \cdots \cup S_j \cup \{p_1\}| < h\frac{\alpha}{2} - h$, we define $X_1 := S_1 \cup \cdots \cup S_j \cup \{p_1\}$. We similarly define $X_2, \cdots, X_k$, as maximal subtrees all rooted at p_1 making sure that $\frac{1}{2}h(\frac{\alpha}{2}-1) < |X_i| < h(\frac{\alpha}{2} - 1)$, with possible exception of X_k which might be of smaller cardinality.

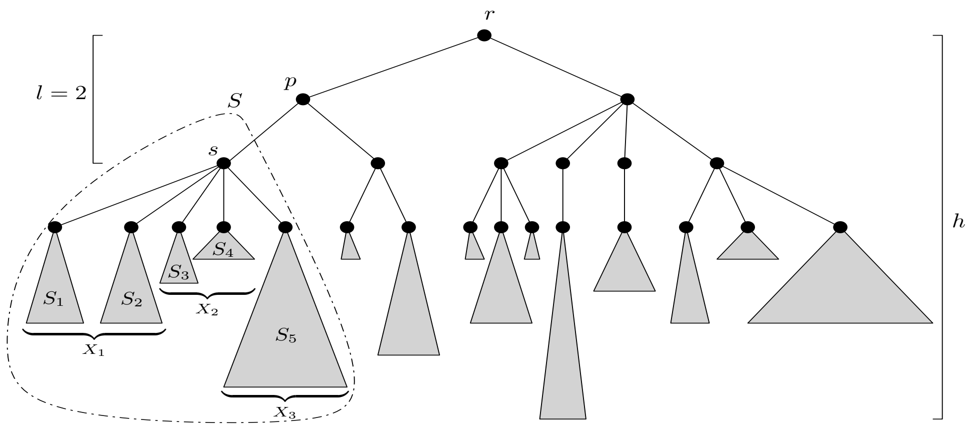

Figure 5.4: Example illustrating grouping of S_i into X_j

For the decontamination process, the agent starts at the root of T, walks its way to p_1 and performs a depth-first search traversal on each X_i one by one. We can afford this because the immunity is strictly greater than the amount of time it takes to perform the traversal on each X_i; in fact its easy to see that any $\tau \geq h\alpha - 2h$ is enough to completely clean P_1.

The next step is to walk up to p_2, the parent of p_1. The plan is to make sure that p_2 never gets recontaminated. Let P_2 be the subtree rooted at p_2. Arbitrarily choose any subtree $R \subseteq P_2 \setminus P_1$, at minimum possible distance from P_2 (e.g., potentially $R = P_2$), with the property that for all subtrees R_i of R, $|R_i| < h\frac{\alpha}{2} - h$ as before. We will group R_i's into X_j's as before but this time after performing a depth-first search traversal on each X_j, we will pay a visit to p_2, making sure it remains decontaminated. Once P_2 is decontaminated, we proceed to p_3 the parent of p_2 and repeat the process until p_j becomes the root of T, and we are done with the decontamination process. From the fact that each X_i is small enough, it is easy to see that $\tau = \alpha h$ is enough for the process. We can always group any tree into at most $2\frac{n}{q}$ subtrees each of size $(h\frac{\alpha}{2} - h) \leq |X_i| \leq h(\alpha - 2)$ where $q = h(\alpha - 2)$. The agent spends at most $2q$ time units on depth first traversal and $2h$ time units on visiting some p_j potentially at distance h for each such subtree. The total amount of time spent in the process is

$$\begin{aligned}
&\leq 2\frac{n}{q} \times 2(q + h) \\
&\leq 2\frac{n}{q} \times 2(q + \frac{q}{\alpha - 2}) \\
&= 2n \times 2(1 + \frac{1}{\alpha - 2}) \\
&= 4n \times \frac{\alpha - 1}{\alpha - 2}
\end{aligned}$$

□

Theorem 6. *Let T be a tree on n vertices, then $\iota(T) = O(\sqrt{n})$.*

Proof. We start with the following observation.

Observation 2. *The decontamination strategy in Lemma 4 is a monotone strategy.*

Now let c be a center vertex of T and let m be the number of leaves in T. Recall that an arm A_i is a set of vertices that lie on the path from c to a leaf l_i for all $1 \leq i \leq m$. Given a tree T rooted at v, we denote by $T_x(v)$ a subtree of T that is attained by removing all vertices from T that are at distance more than x from v i.e, T_x is T truncated at depth x. Assume without loss of generality that leaves l_i are sorted in their depth first search discovery ordering. This implies an ordering on arms A_i. Note that $A_i \setminus \{c\}$ are not disjoint in general. Once we have an order, the agent will start decontaminating arms one by one according to the following algorithm.

- For $i = 1$ to m
  - Perform an auxiliary step and apply Lemma 4 on $T_{\sqrt{n}}(c)$ with $\alpha = 3$.
  - Move the agent from c towards leaf l_i until it reaches a vertex v_j with $deg(v_j) > 2$. We will apply Lemma 4 on $T_{10\sqrt{n}}(v_j)$ again with $\alpha = 3$. After performing an auxiliary decontamination step, we will not perform any more auxiliary steps for next $5\sqrt{n}$ time units of this walk. Since l_i can be at distance at most $\frac{n}{2}$ from c, the total number of auxiliary steps we perform on this walk is bounded above by $\frac{n}{10\sqrt{n}}$. It also follows that no vertex lies in more than two $T_{10\sqrt{n}}(v_j)$'s. We return to c along the shortest path.

To analyze this scheme, we find the following definition useful.

Definition 1. *A vertex v in tree T is called* secured *at some time step i, if it never gets contaminated again.*

The agent decontaminates a new arm A_i in the ith iteration of the algorithm. We observe that

Lemma 5. *The following invariants hold for every step of the algorithm:*

(i) Root c is secured at iteration 1.

(ii) For any secured vertex v, and a contaminated vertex w, which is in same branch as A_i, $dist(v, w) > \sqrt{n}$ at start of iteration $i + 1$.

(iii) All vertices $v_j \in A_i$ are secured at start of iteration $i + 1$.

Proof. We fix $\tau = 30\sqrt{n}$. Let $\Gamma(i)$ be the time spent by the algorithm at iteration i. Then $\Gamma(i)$ can be broken down into three parts: (1) the time spent performing auxiliary decontamination at c, (2) the time spent visiting l_i, and (3) time spent at each auxiliary step on the way to l_i, which is $8a_j$ where a_j denotes size of the tree used in the auxiliary step. We have

$$\begin{aligned}\Gamma(i) &\leq 8n + n + 8\Sigma_j a_j \\ &\leq 8n + n + 16n \\ &= 25n,\end{aligned}$$

where we use the fact that $\Sigma_j a_j$ cannot be more than twice the number of total vertices, since each vertex is used in at most two such auxiliary steps. Since $\tau = 30\sqrt{n}$, after performing an auxiliary decontamination step on tree with $\sqrt{n}$ height, it takes $\geq 30n$ for the contamination to creep back to the root which is less than the time spent in one iteration. This fact along with Observation 2 gives the first invariant (i).

Now a vertex v is secured only if $v \in A_j$ for some $j \leq i$. If v lies in a different branch than the one A_i lies in then invariant (ii), and (iii) follow from (i) i.e. if c is secured then contamination has no way to spread from one branch to another, and if a branch has been decontaminated, it will not get recontaminated. For any contaminated vertex w, $dist(c, w) > \sqrt{n}$ implies that $dist(v, w) > \sqrt{n}$, for any v in a fully decontaminated branch. So we assume without loss of generality that v lies in the same branch as A_i. By the order in which we decontaminate leaves, it is clear that after iteration i, the closest secured vertex to any contaminated vertex, lies in arm A_i. A direct consequence of performing auxiliary decontamination during iteration i is that any contaminated vertex is at distance more than $5\sqrt{n}$ from closest $v \in A_i$. When we have completed iteration i, it is still more than $4\sqrt{n}$ distance away, and this implies invariant (ii).

For any contaminated vertex w, any $u \in A_i$, and any $v \in A_j$, for $j < i$ all contained in the same branch it holds that $dist(u, w) < dist(v, w)$. It follows that $v \in A_j$ for $j < i$ never get contaminated during decontamination process of their branch. This along with (i) implies (iii). □

The claim completes the proof of Theorem with $\iota = 30\sqrt{n}$. □

Remark 3. Although the constant 30 in the proof above can be improved to 6, but the resulting structure of the proof is messier and doesn't yield any further insight into the problem.

5.5 Discussion

While we presented some interesting results, we would like to mention that there are still some very basic questions that seem to be open for further investigations. For example, although we showed that for any tree T, $\iota(T) = O(\sqrt{n})$, it is not clear if this is asymptotically optimal. Using somewhat involved argument, it can be shown that there exist trees T on n vertices for which $\iota(T) = \Omega(n^{\frac{1}{3}+\epsilon})$ for any constant $\epsilon > 0$. It's also noteworthy that if we limit the algorithms to be monotone, its easy to see that $\iota(T) = \Theta(n)$, e.g., consider a spider with three arms of equal length.

Another interesting topology is that of planar graphs. Since meshes are planar graphs, it directly follows from Theorem 5 that

Corollary 4. *There exist planar graphs on n vertices with immunity number $\iota > \frac{\sqrt{n}}{2}$.*

We believe that

Conjecture 4. *Any planar graph G on n vertices can be decontaminated with $\tau(G) = O(\sqrt{n})$.*

The *search number*, $s(G)$, of a graph G is the minimum number of agents needed to decontaminate a graph with $\tau = 0$. In [1], Alon *et al.* proved the following statement, but we give here a simpler, shorter, more intuitive proof.

Theorem 7. *Any planar graph $G = (V, E)$ on n vertices can be decontaminated with $s(G) = O(\sqrt{n})$ agents where vertices of G don't have any immunity.*

Proof. We partition V into three sets V_1, V_2, and S using the Planar Separator Theorem [38] [23], where $|V_i| \leq \frac{2n}{3}$, $|S| \leq 3\sqrt{n}$, and for any $v \in V_1$, and any $w \in V_2$, edge $vw \notin E$. We place $3\sqrt{n}$ agents on S to make sure that contamination can not spread from V_1 to V_2, or vice versa. Let G_1 and G_2 be the subgraphs of G induced on the vertices in V_1 and V_2 respectively. Now let's say it takes $s(G_1)$ agents to decontaminate G_1. Once G_1 is fully decontaminated, we can moving all those agents to decontaminate G_2. Since both G_1 and G_2 are also planar graph, this gives us an obvious recurrence for $s(G)$:

$$s(G) \leq \max(s(G_1), s(G_2)) + 3\sqrt{n}.$$

So the total number of agents required is at most $3\sqrt{n}+3\sqrt{\frac{2n}{3}}+\ldots = O(\sqrt{n})$ which completes the proof. □

Technique used in proof of Theorem 7 may help devise a similar proof for the conjectured bound on immunity number of planar graph. In any case, we do have a hunch that Planar Separator Theorem may be beneficial in that case as well.
Also it's not hard to show that K_n has the maximal immunity number among all graphs on n vertices.

Theorem 8. *Any connected graph $G = (V, E)$ on n vertices can be decontaminated with $\tau = n - 1$.*

Proof. For an arbitrary graph G we present a strategy to decontaminate G with the claimed immunity. Our strategy is a modified depth first search traversal of the graph.

Start with an agent on an arbitrary vertex v_1, and at each time step keep walking the agent to successive *unvisited* neighbors in the depth first fashion. If we exhaust all V then we are done since we visited all vertices before first vertex got recontaminated. Otherwise the agent follows some path $v_1, \ldots, v_{k-1}, v_k$ such that all neighbors of v_k have already been visited. We label a vertex as a *terminal* vertex and plan never to visit it again. So for the rest of the decontamination process, we will assume that v_k does not exist. We traverse the agent back along $v_k, v_{k-1}, \ldots, v_1$ to reach v_1, and then come back along same path to reach v_{k-1}. This time the agent moves to some other neighbor of v_{k-1} if any, and continue as before either finding another another terminal vertex and deleting it too or finding a cycle on rest of the vertices. In either case, process completes in finite time. Since the agent decontaminated the terminal vertices, they cannot contaminate any other vertex after they have been visited. And since, every time the agent encounters a terminal vertex it goes back to v_1, and visits all its neighbors (all of which lie on agent's path back to v_1) in the next less than $n-1$ steps, the terminal vertices cannot get contaminated again. Vertices that are not terminal are decontaminated at the end of the process because they are visited in the traversal on cycle which takes at most $n-1$ steps after we leave v_1. The claim follows. $\square$

This might tempt one to conjecture that $\iota(G)$ is an increasing graph property i.e. if we add new edges to G then the immunity number can only go up. But as the following claim illustrates, this is not the case.

Observation 3. *Immunity number is not an increasing graph property. [32]*

For completeness sake we include the proof here.

Proof. Consider the following counter-example: let G be a spider with $2\sqrt{n}$ arms labeled $A_1, A_2, \ldots, A_{2\sqrt{n}}$, where $|A_i| := \sqrt{n}-1$ when i is even; otherwise A_i has just one vertex.

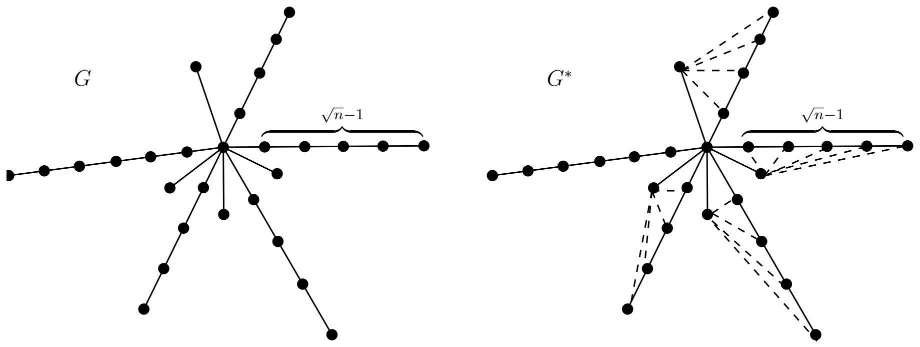

Figure 5.5: (Left) G is a spider tree, and can't be decontaminated with small temporal immunity. We get G^* (on right) by adding dashed edges, and its easy to see that we can decontaminate G^* with $\tau = 2$.

Now construct G^* by adding edges vw where $v \in A_i, w \in A_{i+1}$, for all $i \equiv 1 \pmod 2$ then we can decontaminate G^* with $\tau(G^*) = 2$. We leave it as an exercise for the reader to verify that $\iota(G) > 2$.

$\square$

Other interesting problems related to the topic covered in this chapter include natural generalizations of the problem to directed graphs and weighted graphs. One can also look at the behavior of immunity number of random graphs.

References

[1] Noga Alon and Abbas Mehrabian. Chasing a fast robber on planar graphs and random graphs. *Draft.*, 2013.

[2] Greg Aloupis, Stefan Langerman, Michael Soss, and Godfried Toussaint. Algorithms for bivariate medians and a fermat–torricelli problem for lines. *Computational Geometry*, 26(1):69–79, 2003.

[3] Nina Amenta, Marshall Bern, David Eppstein, and S-H Teng. Regression depth and center points. *Discrete & Computational Geometry*, 23(3):305–323, 2000.

[4] Boris Aronov, Franz Aurenhammer, Ferran Hurtado, Stefan Langerman, David Rappaport, Carlos Seara, and Shakhar Smorodinsky. Small weak epsilon-nets. *Computational Geometry*, 42(5):455–462, 2009.

[5] Imre Bárány. A generalization of carathéodory's theorem. *Discrete Mathematics*, 40(2):141–152, 1982.

[6] Lali Barrière, Paola Flocchini, Pierre Fraigniaud, and Nicola Santoro. Capture of an intruder by mobile agents. In *Proceedings of the Fourteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 200–209. ACM, 2002.

[7] Abdul Basit, Nabil H. Mustafa, Saurabh Ray, and Sarfraz Raza. Hitting simplices with points in $\mathbb{R}^3$. *Discrete & Computational Geometry*, 44(3):637–644, 2010.

[8] Manuel Blum, Robert W Floyd, Vaughan Pratt, Ronald L Rivest, and Robert E Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973.

[9] Béla Bollobás and Imre Leader. Compressions and isoperimetric inequalities. *Journal of Combinatorial Theory, Series A*, 56(1):47–62, 1991.

[10] Béla Bollobás and Imre Leader. Edge-isoperimetric inequalities in the grid. *Combinatorica*, 11(4):299–314, 1991.

[11] Endre Boros and Zoltán Füredi. The number of triangles covering the center of an n-set. *Geometriae Dedicata*, 17(1):69–77, 1984.

[12] Richard Breisch. An intuitive approach to speleotopology. *Southwestern Cavers*, 6(5):72–78, 1967.

[13] Boris Bukh. A point in many triangles. *Journal of Combinatorics*, 13(2):N10, 2006.

[14] Boris Bukh, Jiří Matoušek, and Gabriel Nivasch. Stabbing simplices by points and flats. *Discrete & Computational Geometry*, 43(2):321–338, 2010.

[15] Timothy M Chan. An optimal randomized algorithm for maximum tukey depth. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 430–436. Society for Industrial and Applied Mathematics, 2004.

[16] Fan RK Chung. Pebbling in hypercubes. *SIAM Journal on Discrete Mathematics*, 2(4):467–472, 1989.

[17] Richard Cole, Jeffrey S Salowe, William L. Steiger, and Endre Szemerédi. An optimal-time algorithm for slope selection. *SIAM Journal on Computing*, 18(4):792–810, 1989.

[18] Richard Cole, Micha Sharir, and Chee K Yap. On k-hulls and related problems. *SIAM Journal on Computing*, 16(1):61–77, 1987.

[19] Yassine Daadaa. *Network Decontamination with Temporal Immunity*. PhD thesis, University of Ottawa, 2012.

[20] Yassine Daadaa, Paola cchini, and Nejib Zaguia. Network decontamination with temporal immunity by cellular automata. In *Cellular Automata*, pages 287–299. Springer, 2010.

[21] Yassine Daadaa, Paola Flocchini, and Nejib Zaguia. Decontamination with temporal immunity by mobile cellular automata. In *International Conference on Scientific Computing (CSC)*, pages 172–178, 2011.

[22] Tamal K Dey. Improved bounds for planar k-sets and related problems. *Discrete & Computational Geometry*, 19(3):373–382, 1998.

[23] Hristo Nicolov Djidjev. On the problem of partitioning planar graphs. *SIAM Journal on Algebraic Discrete Methods*, 3(2):229–240, 1982.

[24] Paola Flocchini, Flaminia L Luccio, and L Xiuli Song. Size optimal strategies for capturing an intruder in mesh networks. In *Proceedings of the International Conference on Communications in Computing (CIC), Las Vegas, USA*, pages 200–206, 2005.

[25] Paola Flocchini, Bernard Mans, and Nicola Santoro. Tree decontamination with temporary immunity. In *Algorithms and Computation*, pages 330–341. Springer, 2008.

[26] Jacob Fox, Mikhail Gromov, Vincent Lafforgue, Assaf Naor, and János Pach. Overlap properties of geometric expanders. *Journal für die Reine und Angewandte Mathematik (Crelles Journal)*, 2012(671):49–83, 2012.

[27] Maxime Genest, Jean-Claude Masse, and Jean-Francois Plante. *depth: Depth functions tools for multivariate analysis*, 2012. R package version 2.0-0.

[28] Joseph Gil, William Steiger, and Avi Wigderson. Geometric medians. *Discrete Mathematics*, 108(1):37–51, 1992.

[29] Mikhail Gromov. Singularities, expanders and topology of map. part 2: From combinatorics to topology via algebraic isoperimetry. *Geom. Func. Anal.*, 20:416–526, 2010.

[30] David Haussler and Emo Welzl. ϵ-nets and simplex range queries. *Discrete & Computational Geometry*, 2(1):127–151, 1987.

[31] Shreesh Jadhav and Asish Mukhopadhyay. Computing a centerpoint of a finite planar set of points in linear time. *Discrete & Computational Geometry*, 12(1):291–312, 1994.

[32] Jeff Kahn. Personal communication, 2013.

[33] Daniel Kral, Lukáš Mach, and Jean-Sébastien Sereni. A new lower bound based on Gromovs method of selecting heavily covered points. *Discrete & Computational Geometry*, 48(2):487–498, 2012.

[34] Shay Kutten and David Peleg. Fault-local distributed mending. *Journal of Algorithms*, 30(1):144–165, 1999.

[35] Shay Kutten and David Peleg. Tight fault locality. *SIAM Journal on Computing*, 30(1):247–268, 2000.

[36] Stefan Langerman and William Steiger. Optimization in arrangements. In *STACS 2003*, pages 50–61. Springer, 2003.

[37] Andrea S LaPaugh. Recontamination does not help to search a graph. *Journal of the ACM (JACM)*, 40(2):224–245, 1993.

[38] Richard J Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[39] Regina Liu. A notion of data depth based upon random simplices. *The Annals of Statistics*, 18:405–414, 1990.

[40] Chi-Yuan Lo, Jiří Matoušek, and William Steiger. Algorithms for ham-sandwich cuts. *Discrete & Computational Geometry*, 11(1):433–452, 1994.

[41] Fabrizio Luccio, Linda Pagli, and Nicola Santoro. Network decontamination with local immunization. In *Proceedings of the 20th International Conference on Parallel and Distributed Processing*, pages 264–264. IEEE Computer Society, 2006.

[42] Jiří Matoušek. Computing the center of planar point sets. *Computational Geometry: Papers from the DIMACS special year*, pages 221–230, 1991.

[43] Jiří Matoušek. *Lectures in Discrete Geometry*. Springer-Verlag, New York, NY, 2002.

[44] Nimrod Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM (JACM)*, 30(4):852–865, 1983.

[45] Nimrod Megiddo, S Louis Hakimi, Michael R Garey, David S Johnson, and Christos H Papadimitriou. The complexity of searching a graph. *Journal of the ACM (JACM)*, 35(1):18–44, 1988.

[46] Nabil H Mustafa and Saurabh Ray. An optimal extension of the centerpoint theorem. *Computational Geometry*, 42(6):505–510, 2009.

[47] Nabil H Mustafa, Saurabh Ray, and Mudassir Shabbir. Ray-shooting depth: Computing statistical data depth of point sets in the plane. In *Algorithms–ESA 2011*, pages 506–517. Springer, 2011.

[48] Nabil H Mustafa, Saurabh Ray, and Mudassir Shabbir. k-centerpoints Conjectures for Pointsets in $\mathbb{R}^d$. *Under revision to International Journal of Computational Geometry & Applications*, 2014.

[49] Torrence D Parson. Pursuit-evasion in a graph. theory and applications of graphs. *Lecture Notes in Mathematics, Springer-Verlag*, pages 426–441, 1976.

[50] Torrence D Parsons. The search number of a connected graph. In *Proc. 9th South-Eastern Conf. on Combinatorics, Graph Theory, and Computing*, pages 549–554, 1978.

[51] Ronald Pyke. Spacings. Technical report, DTIC Document, 1965.

[52] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014.

[53] Richard Rado. A theorem on general measure. *Journal of the London Mathematical Society*, 1(4):291–300, 1946.

[54] Géza Tóth. Point sets with many k-sets. *Discrete & Computational Geometry*, 26(2):187–194, 2001.

[55] Uli Wagner. *On k-Sets and Applications*. PhD thesis, ETH Zurich, 2003.