SPATIAL UNCERTAINTY IN MOBILE AND SENSOR NETWORKS

A Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Khuong Vu

December 2012

SPATIAL UNCERTAINTY IN MOBILE AND SENSOR NETWORKS

Khuong Vu

APPROVED:

Rong Zheng, Chairman
Dept. of Computer Science
University of Houston

Jie Gao
Dept. of Computer Science
Stony Brook University

Stephen Huang
Dept. of Computer Science
University of Houston

Olin Johnson
Dept. of Computer Science
University of Houston

Weidong (Larry) Shi
Dept. of Computer Science
University of Houston

Dr. Mark A. Smith, Dean
College of Natural Sciences and Mathematics

Acknowledgements

Working on the Ph.D has been a wonderful and often overwhelming experience. I am indebted to many people for making the time working on my Ph.D an unforgettable experience.

First of all, I am deeply grateful to my advisor, Professor Rong Zheng for her excellent guidance, caring, patience. She has provided me with an excellent atmosphere for doing research; she has oriented and supported me, both financially and spiritually, with promptness and care, and has always been patient and encouraging of new ideas and difficulties; she has been listened to my ideas and discussions no matter it is in late night or early morning. Above all, she has been a steady influence throughout my Ph.D study.

Furthermore, I have been very privileged to get to know and to collaborate with Professor Jie Gao at the department of Computer Science, the State University of New York at Stony Brook. I am grateful to her for hosting my visiting at Stony Brook and providing insightful comments in my work, and for many motivating discussions.

In addition, I am indebted to Professor Rattikorn Hewett at the Texas Tech University. I would not have been in the Ph.D career without her kind help.

I am grateful to the Plonski family, especially Douglas Plonski. They have always loved me as a son in the family, and provided unconditional help during my staying in Stony Brook. My visiting at the Stony Brook University would not have been successful without them.

Finally, and most importantly, I would like to thank my wife, Nguyen. She has always been by my side in all important events of our lives with unconditional

support, encouragement, quiet patience, and love. Her tolerance of my occasional vulgar moods during my Ph.D career is a testament of her unyielding devotion and love. I thank my parents for their faith in me and allowing me to be as ambitious as I wanted.

SPATIAL UNCERTAINTY IN MOBILE AND SENSOR NETWORKS

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Khuong Vu

December 2012

Abstract

Sensor networks have risen in importance in last several years. They have been deployed for several tasks, such as monitoring volcanoes, monitoring buildings and infrastructures, detecting enemy instrution in military, etc... Sensor location plays an important role in network quality. It has impact on different aspects, i.e., network connectivity, network coverage to name a few. However, exact sensor locations are rarely achieved. On the one hand, sensors may be misplaced during operations. On the other hand, sensor locations are kept uncertain due to privacy concerns. This raises the need for investigating sensor networks with the presence of sensor location uncertainty.

This dissertation provides an analysis on sensor networks with the presence of uncertainty. First, we investigate network coverage and target localization and tracking using binary proximity sensors under sensor location uncertainty. A deterministic, polynomial-time algorithm is devised to compute the minimum sensing range for guaranteed coverage. Furthermore, algorithms are proposed for target localization and measurement model for target tracking in sensor networks. The approaches are based on the high order maximum Voronoi diagram of disks in the plane. Next, we study privacy in spatial queries. In contrast to sensor location uncertainty, uncertainty is introduced to spatial queries to protect user information. Essentially, the querying user is grouped with other users to form a cloak for which spatial queries are made, instead of a single query location. In this disseration, a framework for user identity privacy in k-nearest queries is proposed, in which we devise a k-anonymous, locality-preserving cloaking algorithm. Cloaks are also used in spatial skyline queries,

which leads to different domination relationships. The disseration proposes geometric algorithms for spatial skyline queries with user location uncertainty. In addition, the fuzzy domination relationship in spatial skyline queries is investigated. Our work opens up new directions in spatial skyline queries with uncertainty.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

With the advancements in Micro-Electro-Mechanical Systems (MEMS) and sensing technologies, there has been increasing interest in wireless sensor networks (WSNs). A WSN consists of thousands of tiny sensor nodes deployed in a geographical area for observation of an event of interest. These nodes are able to perform sensing, processing, communicating with each other via a wireless ad hoc network to transmit data from different regions of the sensing field. Each node has a transmission range. Node i can receive signal from node j if i is within the transmission range of j. We assume that each node can provides the location information itself, by localization methods such as low-power Global Position System (GPS), or based on signal strengths [13, 8]. Wireless sensor networks can be classified into two classes: *static* and *mobile* based on the positions of wireless nodes. In the former, the position of

nodes does not change, while the nodes in the latter can move quickly. Advances in smart hand-held devices such as smartphone, PDA, and tablets with equipped GPS receivers have promoted the popularity of mobile networks and broaden the concepts of sensor. The user who carries a smartphone may join as a node in a large mobile networks, which have shown huge advantages in various applications such as road monitoring [50, 77] and healthcare [47, 53], to name a few.

Although sensor location is important in mobile and sensor networks, the true sensor locations are rarely achieved, and thus uncertainty always presents. In wireless and mobile sensor networks, even with on-board GPS receivers on all or selected nodes, or execution of distributed location algorithms, there generally exists uncertainty in sensor locations. Even carefully positioned in the deployment phase, sensors may be unintendedly displaced due to environmental or human factors during the course of operation. Furthermore, uncertainty may be introduced on purpose due to privacy and security concerns, i.e., users do not want to reveal their locations. Thus, uncertainty plays both as an unexpected factor, and a mean for privacy protection in mobile and sensor networks. That is to say, uncertainty is a fundamental problem in mobile and sensor networks.

Presence of sensor location uncertainty has impacts on several operational aspects of sensor and mobile networks including link quality, network connectivity, routing, coverage, queries in networks, etc... Much work have been done in literature to model and process uncertainty. A straightforward approach is to estimate sensor locations or to avoid using sensor locations when uncertainty presents. Akoush and Samed use Bayesian learning and neural network to predict mobile node locations in mobile

networks to enhance network connectivity, landmark-based routing schemes [21, 20] employ relative coordinates instead of physical sensor locations. Probabilistic sensing models [84, 30] are widely used to analyze sensor uncertainty where probabilistic models are used to evaluate the detecting probability of sensors. From another perspective, modeling sensor location uncertainty has gained much interest. Much work assumes a Gaussian distribution for sensor locations [85, 42]. Some work assumes the regular hexagonal [34] or identical [9] uncertainty regions of sensor location.

At the same time, uncertainty is closely related to privacy [2]. In the scope of this dissertation, we constraint ourselves to the location privacy in location-based services. This is defined as the "ability to prevent other parties from learning one's current or past location" [10]. Using uncertainty to provide privacy location based services has been intensively investigated in literature. Essentially, user location information are cloaked in a region to hide the true location [27, 10]. However, when users navigate on predefined routes, hiding single locations is not sufficient to protect a user's travel information. Therefore, uncertainty has been utilized to prevent trajectory information leakage [23, 16, 80].

This dissertation concerned with uncertainty using geometric algorithms. We are interested in analyzing sensor networks when sensor locations are associated with uncertainty. We seek for algorithms to analyze the worst case performance of network coverage and target localization and tracking in sensor networks with sensor location uncertainty. Specifically, algorithms are devised for evaluation of the minimum sensing range for guaranteed network k-coverage, target localization, and measurement model in target tracking using sensor networks. The proposed algorithms are

based on the high order maximum Voronoi diagram. In addition, we study uncertainty and its relation to privacy in spatial queries including k-nearest neighbor and spatial skyline queries [63, 66]. While the former is a well-defined and well-studied topic, the spatial skyline query is relatively new. It is a special case of the skyline query [11], which returns "good" results based on domination relationship of objects. Skylie queries, in particular, concern spatial relationships between objects. In this dissertation, a framework for k-nearest neighbor queries in participatory sensing and location-based services is proposed, which provides guaranteed privacy. Additionally, an algorithm is proposed for k nearest neighbor search with cloaks, which is based on the high order Voronoi diagram. Next, we investigate spatial skyline queries where querying locations are given as cloaks. Finally, the spatial skyline query with fuzzy domination relationships is investigated.

1.2 Dissertation organization

We start with the high order maximum Voronoi diagrams of disks in the plane in Chapter 2, the building block for sensor location uncertainty analysis. Then, network coverage and target localization and tracking in binary proximity disk sensor networks is discussed in Chapter 3. Uncertainty in spatial queries is studied in Chapters 4 and 5. Specifically, a k-anonymity privacy framework for k-nearest neighbor queries is proposed in Chapter 4, which presents a k-anonymous, locality-presering cloaking algorithm, and k nearest neighbor search with cloaks. Chapter 5 investigates uncertainty in spatial skyline queries, which deals with user location uncertainty and

fuzzy domination relationships. We conclude the work with the scope of future work in Chapter 6.

Chapter 2

High Order Maximum Voronoi Diagrams of Disks in 2D

We first briefly introduce the concept of Voronoi diagram, and its generalization to the high order Voronoi diagram of points and segments in the plane. Then, we discuss the *maximum* Voronoi diagram of disks in 2D.

2.1 High order Voronoi diagram of points and segments in the plane

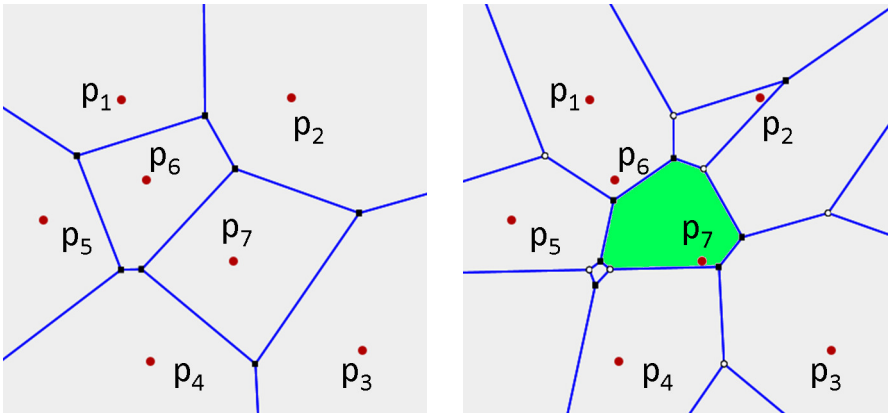
The order-1 Voronoi diagram (Voronoi diagram for short) of a set of points, referred to as sites, in the plane is a tessellation, which divides the plane into a set of regions called Voronoi cells, each corresponding to a site. A Voronoi cell is the locus of points

that are closer to the corresponding site than to the others. The Voronoi diagram of points is formally defined in [7] as follows:

Definition 2.1. *For $p, q \in S$ let $B(p,q) = \{x | d(p,x) = d(q,x)\}$ be the bisector of p and q. $B(p, q)$ is the perpendicular line through the center of the line segment pq. It separates the halfplane $D(p,q) = \{x | d(p,x) < d(q,x)\}$ containing p from the halfplane $D(q,p)$ containing q. We call $VR(p,S) = \bigcap_{q \in S, q \neq p} D(p,q)$ the Voronoi region of p with respect to S. Finally, the Voronoi diagram of S is defined by $V(S) = \bigcup_{p,q \in S, q \neq p} \overline{VR(p,S)} \cap \overline{VR((q,S)}$.*

When we divide the plane into regions, each is the locus of points closer to a set of k sites (instead of a single site) than to the others, we have an order-k Voronoi diagram. Figure 2.1 shows examples of order-1 and order-2 Voronoi diagrams of 7 points. As shown in Figure 2.1b, the shaded area is the order-2 Voronoi cell corresponding to points p_6 and p_7. The distances from any point p in this cell to p_6 and p_7 are smaller than those to other sites. In other words, $d(p, p_6) \leq d(p, p_i)$ and $d(p, p_7) \leq d(p, p_i)$, where $p_i \neq p_6, p_7$, $d(\cdot, \cdot)$ is the Euclidean distance between two points.

Construction of high order Voronoi diagrams requires identifying groups of sites, whose high order Voronoi cells are not empty. In the example shown in Figure 2.1, there exists no point p such that $d(p, p_2) \leq d(p, p_i)$ and $d(p, p_5) \leq d(p, p_i)$, where $p_i \neq p_2, p_5$. Therefore, the order-2 cell corresponding to $\{p_2, p_5\}$ does not exist. In his seminal paper [43], Lee proposed an incremental approach to construct order-k Voronoi diagrams of n points in $O(k^2 n \log n)$ time complexity. To construct the order-k Voronoi diagram, we tessellate every cell of order-$(k-1)$ Voronoi diagram



(a) Order-1 Voronoi diagram.

(b) Order-2 Voronoi diagram. Shaded is the Voronoi cell corresponding to sites p_6 and p_7.

Figure 2.1: An example of high order Voronoi diagrams.

by the order-1 Voronoi diagram of its neighbors. An example is shown in Figure 2.2, which illustrates the tessellation of the example in Figure 2.1a. In this example, the cell corresponding to site p_7 is tessellated by sites p_2, p_3, p_4, and p_6. To tessellate, the order-1 Voronoi diagram of the four neighbors is constructed and superimposed on the cell of p_7. The part of the order-1 Voronoi diagram in the cell is shown by green edges. It is easy to recognize that those edges are parts of the order-2 Voronoi diagram shown in Figure 2.1b.

When sites are line segments, we have Voronoi diagrams of line segments (kSVD for short). The (order-1) Voronoi diagram of line segments is defined as follows:

Definition 2.2 (Segment Voronoi Diagram [7])**.** *The Segment Voronoi Diagram (SVD) is defined over a set of non-intersecting sites, which can either be points*

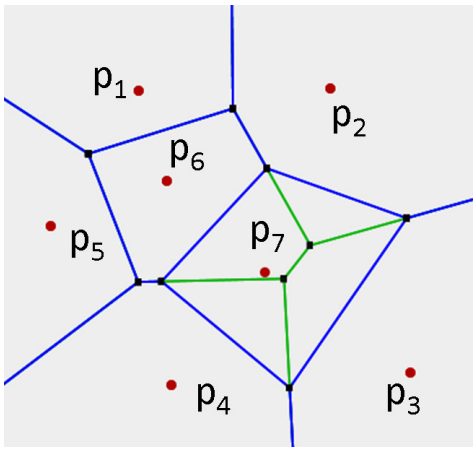


Figure 2.2: Tessellation of the cell corresponding to site p_7 in order-1 Voronoi diagram.

or linear segments. The SVD is a subdivision of the plane into connected regions, called cells, associated with the sites. The cell of a site s_i is the locus of points on the plane that are closer to s_i than any other site s_j, $j \neq i$. The distance $\delta(x, s_i)$ of a point x in the plane to a site s_i is defined as the minimum of the Euclidean distances of x from the points in s_i.

The concept of high order Voronoi diagram of line segments can be similarly adopted from the concept of high order Voronoi diagram of points. An example of Voronoi diagram of line segments is given in Figure 2.3.The distance $d(p, l)$ from a point p to a line segment l is defined as $d(p, l) = \min_{p' \in l} d(p, p')$. Lee's approach to the construction of kVD can be adapted to the construction of kSVD with some modifications. First, a line segment is considered as a set of three sites: two endpoints and the open inner line segment. Second, different from point sites, line segment sites may intersect. A bisector of two sites in kSVD may contain different bisector

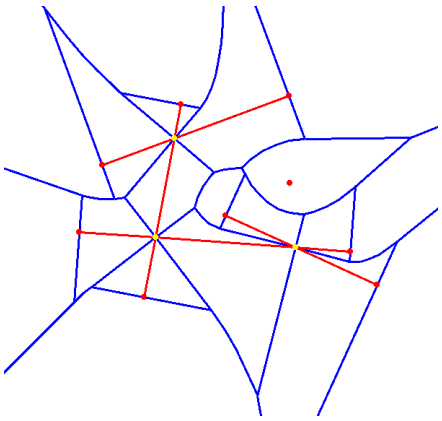


Figure 2.3: An example of Voronoi diagram of line segments. Red lines are line segments. Blue curves are the segment Voronoi diagram.

segments, as shown in Figure 2.4. In this example, the edge of cells of two open line segment l_1 and l_2 consists of the bisector of the inner segments, the bisectors of endpoints and inner line segments, and the bisector of endpoints. As shown in the example, Voronoi edges intersect at the intersection of l_1 and l_2, however, the intersection is not a Voronoi vertex. Roughly speaking, Lee's method can be adapted to construct kSVD of n line segments in $O(k^2 n \log n)$ time complexity.

When the distance from a point p to a point site q is augmented by a positive weight r_q of q, we have the *maximum Voronoi diagram*. Details are given in the following section.

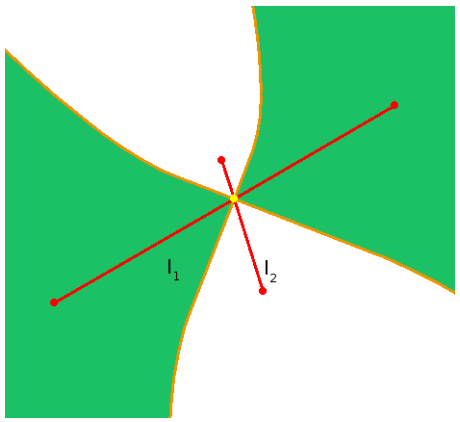


Figure 2.4: Cells and bisectors of kSVD. Shaded is the cell of line segment l_1. Orange curves are the bisectors of the two line segments.

2.2 Maximum Voronoi diagrams of disks

2.2.1 Introduction

Recent years have witnessed an increasing interest in Voronoi diagrams of disks from both theory and application sides. Aurenhammer studied power Voronoi diagrams in [6]. Karavela et al. proposed a dynamic algorithm for additively weighted Voronoi diagrams construction in [36], which allows disk insertion and deletion. [61] studied the guaranteed Voronoi diagram of uncertain sites. In [79], high order Voronoi diagrams of points in 2D are used for moving kNN query authentication. In this paper, we study the high order Voronoi diagram of disks, called *maximum* Voronoi diagrams. We propose a mechanism for updating the diagram when disks change size, which leads to a distributed implementation-friendly algorithm for constructing high order maximum Voronoi diagrams.

In his seminal work [43], Lee proposed an incremental construction algorithm for order-k Voronoi diagrams of points in the plane. In his approach, diagrams of orders from 1 to $k-1$ are constructed to obtained the order-k VD leading to an $O(k^2 N \log N)$ time complexity for N points. Although the approach can be extended to the construction of order-k max VDs of exclusive disks, it does not apply to general cases, i.e., disks may contain other disks. In this paper, we characterize the evolution of high order max VDs as disks expand/shrink. Several advantages stem from this mechanism. First, it allows quick update for high order max VDs as disks' sizes change, resulting in a polynomial complexity algorithm for constructing high order max VD. Second, it enables site insertion and deletion in high order max VDs, which naturally leads to the construction of high order point VDs. Our contributions are as follows:

- A characterization the updates in high order max VDs with disk expansion/shrinking. We show that the expansion of N disks in an order-k max VD of N disks, where each expands once to infinite size, to not contain other disks, takes $O(kN \log N)$.

- We propose an algorithm to order-k max VD construction given the order-k VD of disk centers with $O\left(\left\lceil \frac{r_{\max}-r_{\min}}{d_{\min}} \right\rceil kN \log N\right)$ expected running time complexity, where $r_{\max}$ and $r_{\min}$ are respectively the maximum and minimum radii of N disks, and $d_{\min}$ is the minimum distance between two disk centers.

- We propose a mechanism for site insertion and deletion in order-k max VDs with $O(kN \log N)$ running time complexity where N is the number of disks. It

naturally leads to site insertion and deletion in high order point VDs.

2.2.2 Overview of the proposed approach

Consider a set of N disks $S = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_2(o_2, r_2), \ldots, \mathcal{D}_n(o_N, r_N)\}$ where o_i and r_i are respectively the center and radius of disk $\mathcal{D}_i$ $(1 \leq i \leq N)$. We define the distance from a point p to disk $\mathcal{D}_i$ as $d_{\max}(p, \mathcal{D}_i) = d(p, o_i) + r_i$, where $d(\cdot, \cdot)$ is the Euclidean distance between two points. The locus of points closer to $\mathcal{D}_i$ than to $\mathcal{D}_j$, $h(\mathcal{D}_i, \mathcal{D}_j)$, is one of the half-planes determined by the bisector $b(\mathcal{D}_i, \mathcal{D}_j) = \{p | d_{\max}(p, \mathcal{D}_i) = d_{\max}(p, \mathcal{D}_j)\}$, or $b(\mathcal{D}_i, \mathcal{D}_j) = \{p | d(p, o_i) - d(p, o_j) = r_j - r_i\}$. In general, $b(\mathcal{D}_i, \mathcal{D}_j)$ is a hyperbolic curve. Let $\mathcal{V}(\mathcal{D}_i)$ denote the locus of points closer to $\mathcal{D}_i$ than to any other disk in S. Thus, $\mathcal{V}(\mathcal{D}_i) = \bigcap_{i \neq j} h(\mathcal{D}_i, \mathcal{D}_j)$. It is shown from [36] that if $\mathcal{D}_i$ does not contain any other disk, then $\mathcal{V}(\mathcal{D}_i) \neq \emptyset$. In addition, $\mathcal{V}(\mathcal{D}_i)$'s boundary consists of edges, which are hyperbolic segments, and vertices, which are intersections of adjacent edges. $\mathcal{V}(\mathcal{D}_i)$ is referred to as the Voronoi region (or face) associated with disk $\mathcal{D}_i$, and the set of $\mathcal{V}(\mathcal{D}_i), 1 \leq \ i \leq n$ is referred to as the *maximum Voronoi diagram*, or *max VD* for short, of S. In [36], the authors proposed an algorithm to construct the max VD of n disks in $O(T(N) + N \log N)$, where $T(N)$ is the time of the nearest neighbor query under $d_{\max}$ metric. An example of max VD is shown in Figure 2.5. As seen in the figure, $\mathcal{V}(\mathcal{D}_8) = \emptyset$ since it contains $\mathcal{D}_7$.

Similar to the high order Voronoi diagram of points first studied by Lee [43], we generalize the concept of max VD so as a Voronoi region is associated with a set of disks, $H \subset S$ for $|H| > 1$. Denote $\mathcal{V}^k(H, S)$, where $|H| = k, H \subset S$, the locus of

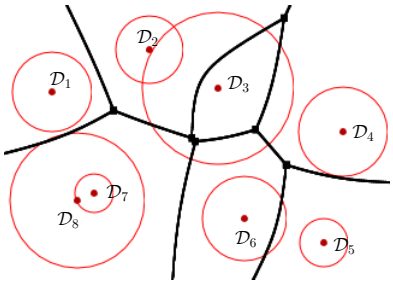


Figure 2.5: The max Voronoi diagram of 8 disks.

points closer to *all* disks of H than to *any* disk in $S \backslash H$. We define the order-k max VD of S, $V^k(S)$, as a collection of Voronoi regions corresponding to all subsets H of S ($|H| = k$), i.e., $V^k(S) = \bigcup_{H \subset S} \mathcal{V}^k(H, S), |H| = k$. We adopt the definition in [7] to formally define the order-k max VD as follows:

Definition 2.3. *For i, $j \in S$, let $D(\mathcal{D}_i, \mathcal{D}_j) = \{p | d_{\max}(p, \mathcal{D}_i) < d_{\max}(p, \mathcal{D}_j)\}$. Let $H \subset S, |H| = k$. We define*

$$\mathcal{V}^k(H, S) = \bigcap_{h \in H, i \notin H} D(\mathcal{D}_h, \mathcal{D}_i)$$

the order-k maximum-Voronoi region of a set of disks H with respect to S. The order-k maximum-Voronoi diagram of S is defined as

$$V^k(S) = \bigcup_{H, H' \subset S; H \neq H'; |H| = |H'| = k} \mathcal{V}^k(H, S) \bigcap \mathcal{V}^k(H', S)$$

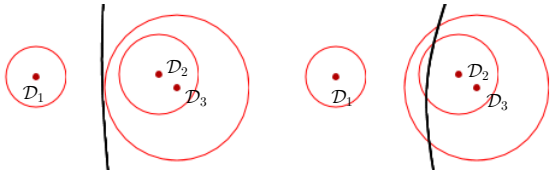
In [73], Lee's incremental algorithm is applied to construct order-k max VD under the assumption that no disk contains any other disk. Accordingly, each Voronoi region of order-$(k-1)$, $\mathcal{V}^k(H, S)$, is tessellated by the order-1 Voronoi diagrams of

some disks to create the next order Voronoi regions. It has been shown that only disks associated with edges of $\mathcal{V}^k(H, S)$ are considered for tessellation. In general placement of disks, the assumption does not always hold. As illustrated in Figure 2.6, although $\mathcal{D}_3$ is associated with no edge in the order-1 max VD of $S = \{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3\}$, $\mathcal{V}^2(\{\mathcal{D}_2, \mathcal{D}_3\}, S) \neq \emptyset$.

We analyze the properties of order-k max VD of disks as disks expand/shrinks. The intuition is as follows. Changing a disk's radius continuously makes some Voronoi vertices move along an identifiable trajectory while the other vertices do not change. We define an *event* as the meeting of two vertices as they are moving. We observe that disks' changes do not necessarily cause an event, that is, a disk expand *continuously* but events happen *discretely*. This is illustrated in Figure 2.7. There are two kinds of vertices, denoted by circles and solid squares. As $\mathcal{D}_7$ expands, vertices of both kinds move along particular edges (arrows in the figures). We observe that solid squares move *away* their corresponding opposite vertex, while circles move *toward* their corresponding opposite vertex. As shown in Figure 2.7b, vertices may meet while moving. In this case, they may "destroy" edge of disks $\mathcal{D}_3$ and $\mathcal{D}_7$, $e_{3,7}$ and create another one, $e_{1,4}$ simultaneously. Additionally, a face may degenerate due to the meeting of moving vertices, e.g., face $\mathcal{V}\{4, 7\}$ of disks $\mathcal{D}_4$ and $\mathcal{D}_7$ in Figure 2.7b. In some cases, another face is born simultaneously, e.g., face $\mathcal{V}\{5, 6\}$ in Figure 2.7b. In other cases, faces degenerate and no face is born, shown in Figure 2.8. In the following sections, we provide details of events that happen when a disk expands. We will see that, the number of events caused by serially expanding n disks in an order-k max VD is $O(kn)$.

Properties of expanding disks are applied to construct order-k max Voronoi diagrams and high order Voronoi diagrams of points. We propose an incremental disk expansion approach to the construction of order-k max VD in which the order-k Voronoi diagram of points is updated to obtain the order-k max VD of disks as disks' radii increase from zero to the given sizes.

In the rest of this chapter, we discuss the update mechanism of high order max VDs as disks expand/shrink is discussed in Section 2.3. The properties of expanding/shrinking disks results in the algorithms for construction and site insertion/deletion in high order max VD in Section 2.4.

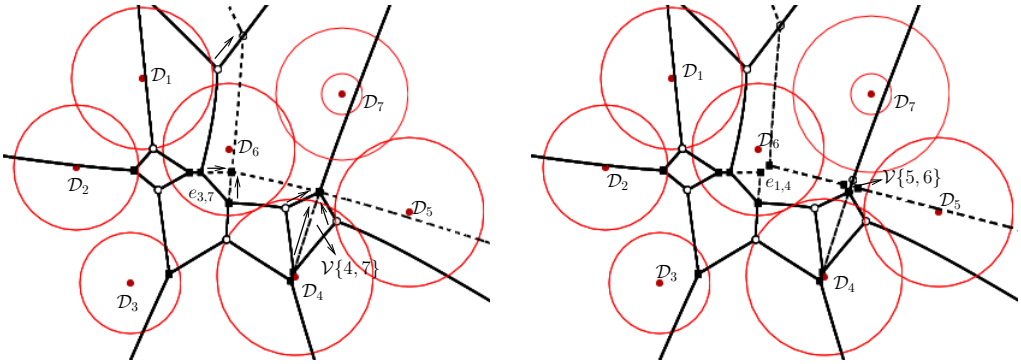


(a) (b)

Figure 2.6: Incremental construction does not apply as disks are contained inside other ones.

2.3 Geometrical analysis of order-k max VD with disk expansion/shrinking

In the following discussion, we assume that no more than 2 disks' centers are co-linear, no point in the plane is equal-distant to more than 3 disks under $d_{\max}$ metric.



(a) (b)

Figure 2.7: Expanding a disk makes edges and/or faces degenerate while creating new edges and/or faces. Dashed edges are corresponding to the expanding disk. Arrows show the movements of vertices.

We summarize the notations used throughout the paper as follows:

$\mathcal{D}_k(o_k, r_k)$**:** The disk centered at o_k with radius r_k. We use the notion $\mathcal{D}_k$ for simplicity when no confusion arises.

S**:** The set of N disks $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_N\}$.

S_i'**:** The modified S, in which the radius of $\mathcal{D}_i$ expands by a positive amount ϵ. $S_i' = (S \backslash \{\mathcal{D}_i\}) \bigcup \{\mathcal{D}_{i'}(o_i, r_i + \epsilon)\}$.

H**:** A subset of S.

H_i'**:** The updated H. $H_i' = (H \backslash \{\mathcal{D}_i\}) \bigcup \{\mathcal{D}_{i'}(o_i, r_i + \epsilon)\}$.

$d_{\max}(p, \mathcal{D}_i)$**:** The maximum distance from p to $\mathcal{D}_i$. $d_{\max}(p, \mathcal{D}_i) = d(p, o_i) + r_i$, where $d(\cdot, \cdot)$ is the Euclidean distance between two 2D points.

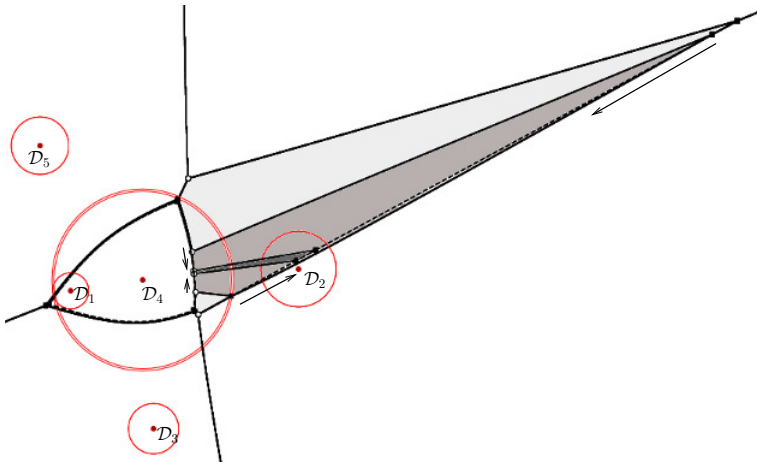


Figure 2.8: The order-2 max VD of 5 disks. As $\mathcal{D}_4$ expands, vertices move as indicated by the arrows. As a result, the darker face shrinks and 2 lighter faces expands. Eventually, the darker face degenerates.

$\mathcal{V}_k(S)$: The max Voronoi region (or face) corresponding to disk $\mathcal{D}_k$ in the max VD of S. Short as $\mathcal{V}_k$ for the same purpose where no confusion occurs.

$V(S)$: The max VD of the disk set S.

$\mathcal{V}^k(H, S)$: An order-k max Voronoi region associated with H, where $|H| = k$.

$V^k(S)$: The order-k max VD of S. When $k = 1$, $V^k(S) \equiv V(S)$.

$v_{i,j,h}$: The max VD vertex corresponding to disks $\mathcal{D}_i$, $\mathcal{D}_j$, and $\mathcal{D}_h$.

$e_{i,j}$: The edge of the max VD corresponding to disks $\mathcal{D}_i$ and $\mathcal{D}_j$. $e_{i,j}$ is a hyperbola segment or an infinite hyperbola.

$b_{i,j}$: The locus of points p such that $d_{\max}(p, \mathcal{D}_i) = d_{\max}(p, \mathcal{C}_j)$. $b_{i,j}$ is a hyperbola with foci being o_i and o_j, or a straight line when $r_i = r_j$. We refer to $b_{i,j}$ as

the bisector of o_i and o_j.

Two circles $\mathcal{D}_1(o_1, r_1)$ and $\mathcal{D}_2(o_2, r_2)$ are internally tangent if $d(o_1, o_2) = |r_1 - r_2|$. In the rest of the paper, by stating that a circle $\mathcal{D}_1$ is internally tangent to $\mathcal{D}_2$, we mean $\mathcal{D}_2$ lies interior to $\mathcal{D}_1$ unless stated otherwise. In addition, a disk $\mathcal{D}_1$ is defined to contain $\mathcal{D}_2$ if $\mathcal{D}_2$ lies interior to $\mathcal{D}_1$ but $\mathcal{D}_1$ is NOT internally tangent to $\mathcal{D}_2$. Consider two edges e_1 and e_2 in a high order max VD. Edge e_2 is defined to *precede* edge e_1, denote $e_1 \prec e_2$, if e_1'target is incident to e_2's source.

2.3.1 Old vertex, new vertex

We review two kinds of vertices described in [43]. Assume p is equal-distant to 3 disks under the $d_{\max}$ metric , i.e., $\mathcal{D}_i, \mathcal{D}_j$, and $\mathcal{D}_q$. Let C be the circle centered at p and internally tangent to the three disks. Assume that C contains $k-1$ other disks. By Definition 2.3, p belongs to 3 Voronoi regions of order k, namely, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i\}, S)$, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j\}, S)$, and $\mathcal{V}^k(H \bigcup \{\mathcal{D}_q\}, S)$. In this case, p is referred to as a vertex of $V^k(S)$. Furthermore,p is also at the intersection of 3 Voronoi regions of order $k+1$, namely, $\mathcal{V}^{k+1}(H \bigcup \{\mathcal{D}_i, \mathcal{D}_j\}, S)$, $\mathcal{V}^{k+1}(H \bigcup \{\mathcal{D}_j, \mathcal{D}_q\}, S)$, and $\mathcal{V}^{k+1}(H \bigcup \{\mathcal{D}_q, \mathcal{D}_i\}, S)$. We say that, p is a *new vertex* of $V^k(S)$ and is an *old vertex* of $V^{k+1}(S)$. In general, a new vertex of order-k Voronoi diagram becomes an old vertex in the order-$(k+1)$ Voronoi diagram; an old vertex of order-k Voronoi diagram is no longer a vertex in the order-$((k+1)$ Voronoi diagram.

We can identify the disks associated with the two edges incident to vertex $v_{i,j,h}$ of face $\mathcal{V}^k(H, S)$ in $V^k(S)$. Let 3 edges incident to $v_{i,j,j}$ are $e_{i,j}$, $e_{i,h}$, and $e_{j,h}$. If $v_{i,j,h}$

is new, and $\mathcal{D}_i \in H$, the 2 edges of $\mathcal{V}^k(H, S)$ incident to $v_{i,j,h}$ must be $e_{i,j}$ and $e_{i,h}$. If $v_{i,j,h}$ is old, and $\{\mathcal{D}_i, \mathcal{D}_j\} \subset H$, the 2 edges of $\mathcal{V}^k(H, S)$ incident to $v_{i,j,h}$ must be $e_{i,h}$ and $e_{j,h}$.

Given an old vertex v of $V^{k+1}(S)$, there is a relationship between the edges of $V^k(S)$ and those of $V^{k+1}(S)$, all incident to v. Let edges of $V^{k+1}(S)$ that are incident to v be $e_{1,2}$, $e_{2,3}$, and $e_{3,1}$. Let C_1 be the circle centered at v and internally tangent to $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$. C_1 contains a set H of $k-1$ disks. We can always move v along $b_{1,2}$, away from the segment that contains $e_{1,2}$ a sufficiently small distance to v', such that circle C_2 centered at v' and internally tangent to $\mathcal{D}_1$ and $\mathcal{D}_2$ still contains H, as illustrated in Figure 2.9. v' must be on an edge of $V^k(S)$. Similar observations can be made for edges $e_{2,3}$ and $e_{3,1}$. We conclude that, there exists an edge $e'_{i,j}$ of order-k max VD incident to the old vertex of edge $e_{i,j}$ of order-$(k+1)$ max VD, which are at the different branch of bisector $b_{i,j}$.

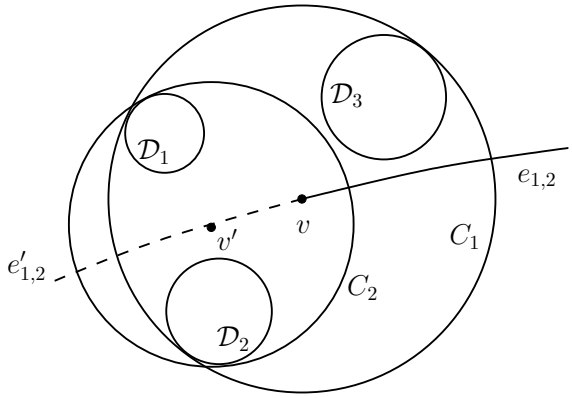


Figure 2.9: Edge $e'_{1,2}$ of order-k max VD (dashed) meets edge $e_{1,2}$ of order-$(k+1)$ max VD (solid) at $e_{1,2}$'s old vertex.

To facilitate our discussion on disk expansion and shrinking, we introduce the notion of *pseudo disk*. A pseudo disk, denotes $\mathcal{D}_\infty$, is defined as a disk centered at the infinity with unit radius. Edges of regular disks and pseudo disks are referred to as *pseudo edges*. Consider the infinite endpoint p of an infinite edge $e_{i,j}$ in an order-1 max VD, we have $d_{\max}(p, \mathcal{D}_j) = d_{\max}(p, \mathcal{D}_i) = \infty$. Therefore, we can associate p with a pseudo disk, that is, p is a vertex corresponding to 3 disks, namely, $\mathcal{D}_j$, $\mathcal{D}_i$, and $\mathcal{D}_\infty$. By this way, we enclose the open end of each order-1 Voronoi region $\mathcal{V}(\{\mathcal{D}_i\}, S)$ with two pseudo edges. Therefore, all regions in any order-1 max VD are considered "closed". By the similar argument, the open end of the infinite edge $e_{i,j}$ is bounded by a new vertex corresponding to disks $\mathcal{D}_i$, $\mathcal{D}_j$, and $\mathcal{D}_\infty$.

We summarize the above discussions in the following propositions.

Proposition 2.1. *A new vertex in order-k max VD of S is an old vertex of order-$(k+1)$ max VD of S.*

Proposition 2.2. *Consider a vertex $v_{i,j,n}$ of $\mathcal{V}^k(H, S)$. If it is new and $\mathcal{D}_i \in H$, the 2 edges of $\mathcal{V}^k(H, S)$ incident to $v_{i,j,n}$ are $e_{i,j}$ and $e_{i,n}$. If it is old and $\{\mathcal{D}_i, \mathcal{D}_j\} \subset H$, the 2 edges of $\mathcal{V}^k(H, S)$ incident to $v_{i,j,n}$ are $e_{i,n}$ and $e_{j,n}$.*

Proposition 2.3. *Consider an old vertex $v_{i,j,n}$ of an order-k max VD, $V^k(S)$. There exists an edge $e'_{i,j}$ of $V^k(S)$. In addition, the edge $e_{i,j}$ $V^k(S)$ in meets $e'_{i,j}$ at $v_{i,j,n}$ but they do not overlap.*

Proposition 2.4. *The open end of an infinite edge $e_{i,j}$ of $V^k(S)$ can be viewed as a new vertex of $V^k(S)$ corresponding to 3 disks, namely, $\mathcal{D}_i, \mathcal{D}_j$, and $\mathcal{D}_\infty$.*

We now describe the structure of $V^k(S)$ by adopting the results in [43], with

extension to new and old vertices.

Proposition 2.5. *Consider a circle C expanding from a point p in the plane contains $(k-1)$ disks, $\mathcal{D}_1, \ldots, \mathcal{D}_{k-1}$. If C is internally tangent to exactly one disk $\mathcal{D}_k$, then $p \in \mathcal{V}^k(\{\mathcal{D}_1, \ldots, \mathcal{D}_k\}, S)$. If C is internally tangent to exactly 2 disks, $\mathcal{D}_k$ and $\mathcal{D}_{k+1}$, then $p \in e_{k,k+1}$ of $\mathcal{V}^k(\{\mathcal{D}_1, \ldots, \mathcal{D}_k\}, S)$ and $\mathcal{V}^k(\{\mathcal{D}_1, \ldots, \mathcal{D}_{k-1}, \mathcal{D}_{k+1}\}, S)$. If C is internally tangent to 3 disks, p is a new vertex in $V^k(S)$, and, simultaneously, an old vertex in $V^{k+1}(S)$.*

The result implies that the order-k max VD is a tessellation of the plane, which divides it into regions, each of which is a locus of points closer to a set of k disks than to other disks. There may be several non-contiguous regions corresponding to an particular set of k disks in an order-k max VD when $k > 1$. Furthermore, some disks may not contribute any edge in the diagram, as illustrated in the example in Figure 2.6. Generally speaking, disk expansion and shrinking are two opposite processes. Therefore, we only consider one for convenience, the other one is similar. In this work, we study the evolution of the order-k max VD as a disk expands. More specifically, we investigate 2 cases, namely, *i*) the disk that shares edges with at least one disk, and *ii*) the disk that does not share edges with any disk. We refer the disks of the first case as *type-I*, and the latter as *type-II*. Expanding a type-II disk eventually makes it a type-I, while expanding a type-I disk possibly makes some type-I disks type-II. In the following sections, we establish fundamental properties as a disk in an order-k max VD expands. We study type-I disks in section 2.3.2 and discuss in section 2.3.3 the expansion of a type-II disk.

The main result of the work is as follows: as disks expand, events happen. However, when each disk expands (possibly to infinite size) once, the number of events is bounded by $O(kN)$. Formally,

Theorem 2.1. *Consider an order-k max VD of N type-I disks, where $k < N$. The expected number of events happening as each disk expands once, either to infinite size, or to not contain other disks, is $O(kN)$.*

We first go over the foundation before sketching the proof. We start by defining the *maximum circumference*. Consider an edge $e_{i,j}$ of 2 regions $\mathcal{V}^k(H_1, S)$, $\mathcal{V}^k(H_2, S)$. By Proposition 2.5, there exists a circle internally tangent to $\mathcal{D}_i$ and $\mathcal{D}_j$, and contains $H_1 \bigcup H_2$. We refer to the circle as the *maximum circumference of H_1 and H_2* associated with $e_{i,j}$, or simply maximum circumference when it is clear in the context. Similarly, a maximum circumference associated with a vertex is defined as the circle centered at the vertex and internally tangent to the disks constructing the vertex. When $e_{i,j}$ is an infinite edge, there exists a maximum circumference centered at infinity.

2.3.2 Expansion of a type-I disk

Expanding a disk $\mathcal{D}_i$, where $\mathcal{D}_i \in H$, makes some regions $\mathcal{V}^k(H, S)$ shrink.

Lemma 2.1. *Let $\mathcal{V}^k(H, S)$ and $\mathcal{V}^k(H'_i, S'_i)$ be the max Voronoi region of H and H'_i in S and S'_i, respectively. Then, $\mathcal{V}^k(H'_i, S'_i) \subset \mathcal{V}^k(H, S)$.*

Proof. Let $p \in \mathcal{V}^k(H'_i, S'_i)$. By definition, $d_{\max}(p, \mathcal{D}_j) \geq d_{\max}(p, \mathcal{D}_q)$, $\forall j \in S, \forall q \in H'_i$.

Since H_i' only differs from H by i in that $r_{i'} > r_i$, we have $d_{\max}(p, \mathcal{D}_j) \geq d_{\max}(p, \mathcal{D}_q)$, $\forall j \in S$, $\forall q \in H$. Therefore, $p \in \mathcal{V}^k(H, S)$. □

Roughly speaking, expanding a disk leads to changes in vertices, edges, and regions of $V^k(S)$. As a high order max VD evolves, different sequences of events occur as different types of vertices and edges are involved. Vertices move along edges and meet other ones. The meeting of 2 new vertices or 2 old vertices results in an edge-death/birth, while the meeting of an old vertex and a new one additionally leads to face-death/birth.

We first characterize the changes in vertices when a disk expands in the following lemma.

Lemma 2.2. *Consider a vertex $v_{i,j,q}$ of $\mathcal{V}^k(H, S)$. As $\mathcal{D}_i$ expands, $v_{i,j,q}$ moves along $b_{j,q}$. If $v_{i,j,q}$ is new, it moves away from the other end of $e_{j,q}$ elongating $e_{j,q}$. Otherwise, it moves towards the other end of $e_{j,q}$ to shorten $e_{j,q}$.*

Proof. We first prove the case of new vertices. W.l.o.g, assume $\mathcal{D}_i \in H$. The circle centered at $v_{i,j,q}$ and internally tangent to $\mathcal{D}_i$, $\mathcal{D}_j$, and $\mathcal{D}_q$ contains $H \backslash \{\mathcal{D}_i\}$. Since no point in the plane is equal-distant to more than 3 disks, there exists a sufficiently small ϵ such that there exists a circle C internally tangent to $\mathcal{D}_{i'}(o_i, r_i + \epsilon)$, $\mathcal{D}_j$, and $\mathcal{D}_q$, and contains $H \backslash \{\mathcal{D}_i\}$. Let $v_{i',j,q}$ be the center of C. Clearly, $v_{i',j,q} \in b_{j,q}$. Since C is internally tangent to $\mathcal{D}_{i'}$, it contains $\mathcal{D}_i$. Therefore, it must not be in the ray part of $b_{j,q}$ originated at $v_{i,j,q}$, which contains $e_{j,q}$. This implies that $e_{j,q}$ is elongated. If $v_{i,j,q}$ is old, it is a new vertex of the order $k-1$ diagram from earlier discussions. Moreover, edge $e'_{j,q}$ of order $k-1$ and edge $e_{j,q}$ of order k diagrams are both incident

to $v_{i,j,q}$, but lie on different halves of $b_{j,q}$. As $\mathcal{D}_i$ expand, $v_{i,j,q}$ moves to elongate $e'_{j,q}$, which equivalently shortens $e_{j,q}$. □

The lemma establishes the motion of different kinds of vertices, i.e., old vertices and new vertices, as the result of expansion of an associated disk, Essentially, motion of old vertices in an order-k max-VD is equivalent of their motion in the order-$(k-1)$ max-VD. This is illustrated in Figure 2.10.

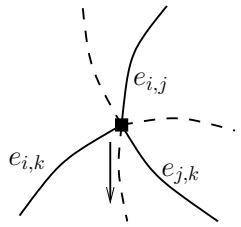


Figure 2.10: Illustration of a vertex's movement. Solid curves are edges of the order-$(k-1)$ max VD. Dashed curves are edges of the order-k max VD, which are the extension of the corresponding edges in the order-$(k-1)$ max VD. The square is a new vertex $v_{i,j,k}$ of the order-$(k-1)$ max VD. As $\mathcal{D}_i$ expands, $v_{i,j,k}$ moves along $e_{i,j}$ away from the other ends of $e_{i,j}$ in the order-$(k-1)$ max VD, and toward the other end of $e_{i,j}$ in the order-k max VD (shown by the arrow).

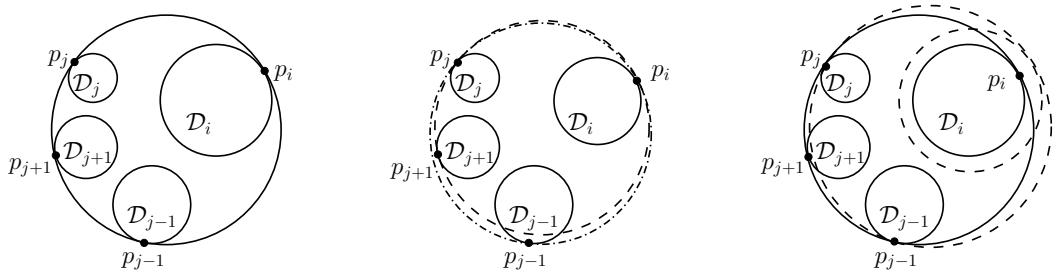
Vertices of an order-k max-VD meet when disks expand leading to events. When different types of vertex meet, different consequences entail. We discuss them in details shortly. It is important to note that, $b_{j,q}$ is static irrespective of the expansion of $\mathcal{D}_i$ even if the expansion of $\mathcal{D}_i$ may lead to the movement of both ends of $e_{j,q}$. This is useful in evaluating the expansion amount of $\mathcal{D}_i$ when meetings of vertices

happen. Given a disk $\mathcal{D}_i$ and the corresponding order-k max VD, we can always evaluate the maximum expansion of $\mathcal{D}_i$ till the next event happens.

Lemma 2.3. *Let $e_{i,j}$ be an edge of $\mathcal{V}^k(H,S)$ with 2 new vertices, e.g., $v_{i,j-1,j}$ and $v_{i,j,j+1}$ in counter-clockwise order in $\mathcal{V}^k(H,S)$. $e_{i,j-1}$ and $e_{i,j+1}$ are the edges of $\mathcal{V}^k(H,S)$ incident to $v_{i,j-1,j}$ and $v_{i,j,j+1}$, respectively. Let p be the intersection of $b_{j,j-1}$ and $b_{j,j+1}$. Assume that the next event as $\mathcal{D}_i$ expands is the meeting of $v_{i,j-1,j}$ and $v_{i,j,j+1}$ at p. If $\mathcal{D}_{j-1} \neq \mathcal{D}_{j+1}$, then with further expansion of $\mathcal{D}_i$, $e_{i,j} = \emptyset$ and $e_{j-1,j+1} \neq \emptyset$. In addition, both vertices of $e_{j-1,j+1}$ are new.*

Proof. Let $\mathcal{D}_{i'} := \mathcal{D}_i(o_i, r_i + d_{\max}(p, \mathcal{D}_j))$, i.e., $\mathcal{D}_i$ expands to $\mathcal{D}_{i'}$ such that $v_{i,j-1,j}$ meets $v_{i,j,j+1}$ at p. The maximum circumference C centered at p, is internally tangent to disks $\mathcal{D}_{j+1}$, $\mathcal{D}_j$, $\mathcal{D}_{j-1}$, and $\mathcal{D}_i$. Since $v_{i,j-1,j}$ and $v_{i,j,j+1}$ are new vertices, C contains $H\backslash\{\mathcal{D}_i\}$. Let p_{j+1}, p_j, and p_i be the tangent point of disks $\mathcal{D}_{j+1}$, $\mathcal{D}_j$, and $\mathcal{D}_i$ with C. Let p_ip_j, p_ip_{j+1}, and p_jp_{j+1} be the arcs of C such that they do not contain p_{j-1}, p_j, and p_i, respectively. Let p_{j-1} be the tangent point of $\mathcal{D}_{j-1}$ with C. We claim that $p_{j-1} \notin p_ip_{j+1} \bigcup p_jp_{j+1}$. We prove by contradiction. Assume $p_{j-1} \in p_ip_{j+1}$ as shown in Figure 2.11a. Since $\mathcal{D}_{i'}$ contains $\mathcal{D}_i$, the maximum circumference centered at p must contain $\mathcal{D}_{j+1}$ as shown in Figure 2.11b, which is a contradiction. By similar arguments, we can show that $p_{j-1} \notin p_jp_{j+1}$. Therefore, $p_{j-1} \in p_ip_j$. As $\mathcal{D}_i$ expands further from $\mathcal{D}_{i'}$, all disks that contain $\mathcal{D}_j$ and $\mathcal{D}_i$ must contain $\mathcal{D}_{j+1}$ or $\mathcal{D}_{j-1}$ (Figure 2.11c). Therefore, edge $e_{i,j}$ does not exist in $\mathcal{V}^k(H,S)$. Furthermore, C still contains $H\backslash\{\mathcal{D}_i\}$ and is internally tangent to $\mathcal{D}_j$, $\mathcal{D}_{j-1}$, and $\mathcal{D}_{j+1}$. Since $|H\backslash\{\mathcal{D}_i\}| = k-1$, $v_{j-1,j,j+1}$ is a new vertex of $V^k(S)$, and thus, there exists an edge $e_{j-1,j+1}$ of $V^k(S)$. Moreover, with a sufficiently small expansion of $\mathcal{D}_i$ from $\mathcal{D}_{i'}$ there

exists a disk that is internally tangent to $\mathcal{D}_i$, $\mathcal{D}_{j-1}$, and $\mathcal{D}_{j+1}$ and contains $H \backslash \{\mathcal{D}_i\}$. Thus, $v_{i,j-1,j+1}$ is a new vertex of $e_{j-1,j+1} = \emptyset$. This completes the proof. □



(a) When $v_{i,j-1,j} \equiv v_{i,j,j+1}$, $\mathcal{D}_i$, $\mathcal{D}_j$, $\mathcal{D}_{j-1}$, and $\mathcal{D}_{j+1}$ are co-circular.

(b) As $\mathcal{D}_i$ shrinks, the maximum circumference centered at $v_{i,j,j+1}$ (dashed) does not contain $\mathcal{D}_{j-1}$ but that centered at $v_{i,j,j-1}$ (dot-dashed) contains $\mathcal{D}_{j+1}$.

(c) With a sufficiently small expansion of $\mathcal{D}_i$, the maximum circumference internally tangent to $\mathcal{D}_i$, $\mathcal{D}_{j+1}$, and $\mathcal{D}_{j-1}$ (dot-dashed) does not contain $\mathcal{D}_j$.

Figure 2.11: Illustration of Lemma 2.3's proof.

Lemma 2.4. *Consider an edge $e_{i,j}$ of 2 regions $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$ with 2 old vertices $v_{i,j,n}$ and $v_{i,j,m}$, where $\{\mathcal{D}_i, \mathcal{D}_n, \mathcal{D}_m\} \subset H_1$ and $\{\mathcal{D}_j, \mathcal{D}_n, \mathcal{D}_m\} \subset H_2$, respectively. Let $\mathcal{V}^k(H_3, S)$ and $\mathcal{V}^k(H_4, S)$ be the other regions, incident to $v_{i,j,n}$ and $v_{i,j,m}$, respectively. If $\mathcal{D}_m \neq \mathcal{D}_n$, and $\mathcal{D}_n$ expands such that the next event is the meeting of $v_{i,j,n}$ and $v_{i,j,m}$, further expansion of $\mathcal{D}_n$ results in i) $e_{i,j} = \emptyset$, and ii) "new" edge $e_{n,m} \neq \emptyset$ of $\mathcal{V}^k(H_3, S)$ and $\mathcal{V}^k(H_4, S)$.*

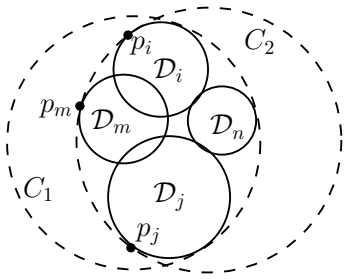


Figure 2.12: Circle C_1 is internally tangent to $\mathcal{D}_i, \mathcal{D}_j$, and $\mathcal{D}_n$ at i_1, j_1, and n_1, respectively, and contains $\mathcal{D}_m$. Circle C_2 is internally tangent to $\mathcal{D}_i, \mathcal{D}_j, \mathcal{D}_m$ and contains $\mathcal{D}_n$. As $\mathcal{D}_n$ expands, it must first touch C_1 at arc $j_1 i_1$

Proof. Let C_1 and C_2, respectively, be the circles centered at $v_{i,j,m}$ and $v_{i,j,n}$, and internally tangent to 2 set of circles $\{\mathcal{D}_i, \mathcal{D}_j, \mathcal{D}_m\}$ and $\{\mathcal{D}_i, \mathcal{D}_j, \mathcal{D}_n\}$. By the property of old vertices, C_1 contains $\mathcal{D}_n$, and so does C_2 for $\mathcal{D}_m$. Let p_i, p_j, and p_m be the tangent points of C_1 with $\mathcal{D}_i, \mathcal{D}_j$, and $\mathcal{D}_m$, respectively. Let $p_j p_i$, $p_i p_m$, and $p_m p_j$, respectively, be the arcs that does not contains p_m, p_j, and p_i. When $\mathcal{D}_n$ expands, it may first touch C_1 at arcs $p_j p_i$, $p_i p_m$, or $p_m p_j$. Since when $v_{i,j,n} \equiv v_{i,j,m}$, C_1 and C_2 coincide and both tangent to $\mathcal{D}_i, \mathcal{D}_j, \mathcal{D}_m$, and $\mathcal{D}_n$, $\mathcal{D}_n$ must first touch C_1 at $p_j p_i$ by the similar arguments as in the proof of Lemma 2.3. This is illustrated in Figure 2.12. Moreover, as $\mathcal{D}_n$ expands further, there is no circle internally tangent to $\mathcal{D}_i$ and $\mathcal{D}_j$ that contains $\mathcal{D}_m$ and $\mathcal{D}_n$. Therefore, edge $e_{i,j}$ of $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$ disappears. Moreover, there exists a circle C internally tangent to $\mathcal{D}_m$ and $\mathcal{D}_n$ that contains $\mathcal{D}_i$ and $\mathcal{D}_j$. In addition, C contains C_1, and thus, C contains all disks of H_3 and H_4. Therefore, there exists an edge $e_{m,n}$ of $\mathcal{V}^k(H_3, S)$ and $\mathcal{V}^k(H_4, S)$ due to Proposition 2.5. Furthermore, both the vertices $v_{i,n,m}$ and $v_{j,n,m}$ of $e_{m,n}$ are old. □

Lemmas 2.3 and 2.4 establish the changes in the diagram as vertices of the same kind meet. In general, meeting of the 2 ends of an edge makes the edge disappear. We refer to this as an *edge-death*. If the two vertices differ by one associated disk, another edge is born simultaneously. We refer to this as an *edge-birth*. Vertices of the newly born edge are new or old depending on the type of the meeting vertices. This is illustrated in Figures 2.14 and 2.15. We next study the changes in the diagram if the two vertices are associated with the same set of disks. We first consider an example in Figure 2.13, which shows an order-1 max VD of 3 disks $\mathcal{D}_1, \mathcal{D}_2$, and $\mathcal{D}_3$. Both vertices, $v_{1,2,3}$ in the example are associated with the same 3 disks. As $\mathcal{D}_2$ expands, the two vertices move closer, and the face of $\mathcal{D}_2$ shrinks as shown in Figure 2.13a. Eventually, two vertices meet, and $\mathcal{D}_2$'s face disappears, as shown in Figure 2.13b. Another example is shown in Figure 2.8, in which expansion of disk $\mathcal{D}_4$ leads to a face-death as it contains $\mathcal{D}_1$.

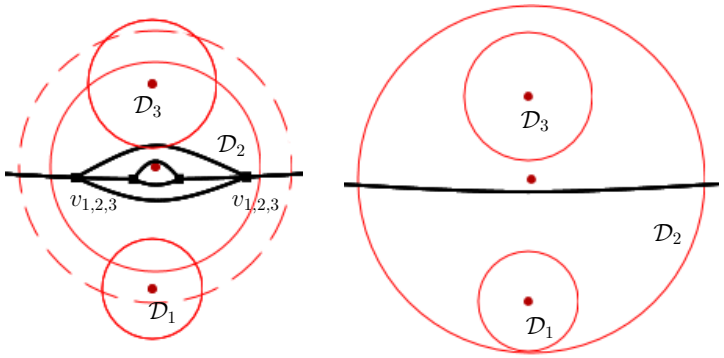


Figure 2.13: The order-1 max VD of 3 disks. Face of disk $\mathcal{D}_2$ (Figure a) disappears as 2 vertices associated with the same sites meet due to the expansion of disk $\mathcal{D}_2$ (Figure b).

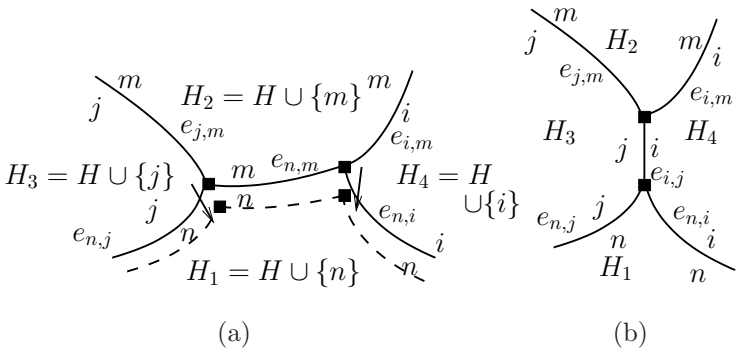


Figure 2.14: The evolution of edges $e_{i,j}$ and $e_{n,m}$ as $\mathcal{D}_n$ expands. Initially, $e_{n,m} \neq \emptyset$, and $e_{i,j} = \emptyset$ (Figure a). As $\mathcal{D}_n$ expands, 2 vertices of $e_{n,m}$ moves along the corresponding edges toward $\mathcal{V}^k(H_1, S)$ (arrow). If they meet, $e_{n,m}$ disappears and $e_{i,j}$ is born (Figure b). Both $e_{i,j}$'s vertices are new.

Lemma 2.5. *Let $v^1_{n,m,i}$ and $v^2_{n,m,i}$ be two distinct new vertices of $\mathcal{V}^k(H, S)$ ($\mathcal{D}_n \in H$, $\{\mathcal{D}_m, \mathcal{D}_i\} \not\subset H$). Assume that the next event as $\mathcal{D}_n$ expands is the meeting of $v^1_{n,m,i}$ and $v^2_{n,m,i}$ (and thus, $e_{n,m}$ disappears). Let $\mathcal{D}_{n'}(o_n, r_{n'})$ be the expansion of $\mathcal{D}_n$ when the event happens. If $\mathcal{D}_n$ expands further from $\mathcal{D}_{n'}$, $\mathcal{V}^k(H, S) = \emptyset$. Furthermore, there exists a sufficient small ϵ such that as $\mathcal{D}_n$ shrinks by ϵ from $\mathcal{D}_{n'}$, $\mathcal{V}^k(H, S)$ consists of 4 edges, namely, $e_{n,m}$, $e_{n,ni}$, $e_{q,i}$, and $e_{n,i}$ for some $\mathcal{D}_q \in H$ if $\mathcal{D}_n$ contains no more than $k-1$ disks; otherwise, $\mathcal{V}^k(H, S)$ consists of 2 edges, namely, $e_{n,m}$ and $e_{n,i}$.*

Proof. Since $v^1_{n,m,i}$ and $v^2_{n,m,i}$ are two distinct new vertices, there are 2 distinct maximum circumferences centered at $v^1_{n,m,i}$ and $v^2_{n,m,i}$ and both internally tangent to $\mathcal{D}_n$, $\mathcal{D}_m$, and $\mathcal{D}_i$. As $v^1_{n,m,i}$ meets $v^2_{n,m,i}$ they coincide. This happens when $\mathcal{D}_n$ is internally tangent to either $\mathcal{D}_i$ or $\mathcal{D}_m$, or both. We first show that if $e_{n,m}$ disappears, $\mathcal{D}_n$ must

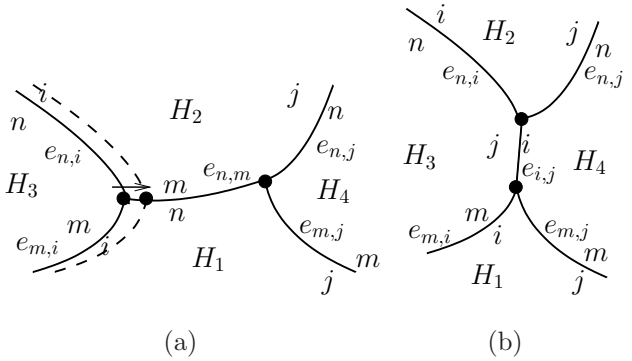


Figure 2.15: The evolution of edges $e_{n,m}$ and $e_{i,j}$ as $\mathcal{D}_i$ expands. Initially, $e_{i,j} = \emptyset$ and $e_{n,m} \neq \emptyset$ (Figure a). As $\mathcal{D}_i$ expands, a vertex of $e_{n,m}$ moves toward the opposite end (arrows), which makes $e_{n,m}$ shrinks. Eventually, $e_{n,m}$ degenerates as 2 vertices of $e_{n,m}$ meet, and $e_{i,j}$ is born (Figure b). Both $e_{i,j}$'s vertices are old.

be internally tangent to $\mathcal{D}_i$ as the event happens. We prove by contradiction. Let C be the maximum circumference centered at $v^1_{n,m,i} \equiv v^2_{n,m,i}$. C contains $H \backslash \{\mathcal{D}_n\}$. As $v^1_{n,m,i}$ meets $v^2_{n,m,i}$, edge $e_{n,m}$ degenerates into a point. Assume $\mathcal{D}_n$ is internally tangent to $\mathcal{D}_m$, we can shrink C by a sufficiently small amount (and move the center of C as well) such that it is still internally tangent to $\mathcal{D}_m$ and $\mathcal{D}_n$ at their contact point. C now does not contain $\mathcal{D}_i$ but still contains $H \backslash \{\mathcal{D}_n\}$. This means that there exist *more* than one point on $e_{n,m}$, which is a contradiction. Now, as $\mathcal{D}_n$ expands further, any disk containing $\mathcal{D}_n$ must contain $\mathcal{D}_i$. Since $\mathcal{D}_i \notin H$, $\mathcal{V}^k(H, S) = \emptyset$. This proves the first claim. Next, as $\mathcal{D}_n := \mathcal{D}_{n'}(o_i, r_{i'} - \epsilon)$, $\mathcal{V}^k(H, S) \neq \emptyset$ and $e_{n,m} \neq \emptyset$. We show that if $\mathcal{D}_n$ contains no more than $k-2$ disks, $\exists \mathcal{D}_q$ such that $e_{q,i} \neq \emptyset$. This can be accomplished by shrinking C such that it is still internally tangent to $\mathcal{D}_n$ and $\mathcal{D}_i$. As C contains $H \backslash \{\mathcal{D}_n\}$ and $\mathcal{D}_n$ contains no more than $k-2$ disks, C will eventually

be internally tangent to some $\mathcal{D}_q \in H$. Therefore, we can shrink $\mathcal{D}_n$ by a sufficiently small amount such that there are 2 disks which are internally tangent to $\mathcal{D}_i$, $\mathcal{D}_n$, and $\mathcal{D}_q$, and contains $H \backslash \{\mathcal{D}_n\}$. This shows that $e_{q,i} \neq \emptyset$. Additionally, both vertices of $e_{q,i}$ are corresponding to 3 disks $\mathcal{D}_i$, $\mathcal{D}_n$, and $\mathcal{D}_q$. Therefore there exist 2 edges $e_{n,i}$ incident at the 2 vertices of $e_{q,i}$. Furthermore, $\mathcal{D}_n$ can always shrink by a sufficiently small amount such that any maximum circumference associated with any disk of H must contain $\mathcal{D}_i$. In the case $\mathcal{D}_n$ contains $k-1$ disks, such a $\mathcal{D}_q$ does not exist. Thus, $\mathcal{V}^k(H, S)$ only contains $e_{n,i}$ and $e_{n,m}$. This proves the latter claim. □

The lemma shows that a face degenerates due to the meeting of 2 new vertices associated with the same 3 disks. We refer to this as a *face-death*. It also provides the face's structure before it degenerates due to the meeting of 2 vertices of the same three disks. Using the similar arguments, we also observe face-death as 2 old vertices of the same 3 disks meets as stated in Lemma 2.6.

Lemma 2.6. *Let $v^1_{n,i,j}$ and $v^2_{n,i,j}$ be two distinct old vertices of $\mathcal{V}^k(H, S)$, where $\{\mathcal{D}_n, \mathcal{D}_i\} \subset H$. Assume that the next event as $\mathcal{D}_n$ expands is the meeting of $v^1_{n,i,j}$ and $v^2_{n,i,j}$ which make $e_{i,j}$ degenerate. Let $\mathcal{D}_{n'}(o_i, r_{i'})$ be the expansion of $\mathcal{D}_n$ when the event happens. As $\mathcal{D}_n$ expands further from $\mathcal{D}_{n'}$ either region corresponding to $e_{i,j}$, e.g., $\mathcal{V}^k(H, S) = \emptyset$. Furthermore, there exists a sufficient small ϵ such that as $\mathcal{D}_n$ shrinks by ϵ from $\mathcal{D}_{n'}$, $\mathcal{V}^k(H, S)$ consists of 4 edges, namely, $e_{i,j}$, $e_{n,j}$, $e_{n,q}$, and $e_{n,j}$ (for some $\mathcal{D}_q \notin H$).*

Next, we turn to the discussion of the meeting of vertices of different kinds. Roughly speaking, when a new vertex meets an old one due to the expansion of

some disk, some region degenerates causing a face-death, and a new face comes to existence simultaneously. We refer to this as a *face-birth*. We formally present the results as follows.

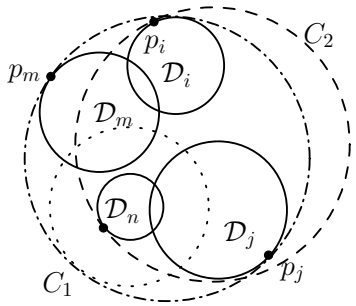


Figure 2.16: If C_2 (dashed) does not contain $\mathcal{D}_m$, while C_1 (dot-dashed) contains $\mathcal{D}_n$ then as $\mathcal{D}_n$ expands (dotted), it is first tangent to C_1 at arc $p_m p_j$ or $p_i p_m$.

Lemma 2.7. *Consider an edge $e_{i,j}$ with an old vertex $v_{i,j,n}$ and a new vertex $v_{i,j,m}$ of 2 regions $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$, where $\{\mathcal{D}_i, \mathcal{D}_n\} \subset H_1$ and $\{\mathcal{D}_j, \mathcal{D}_n\} \subset H_2$. If $\mathcal{D}_n$ expands such that the next event is the meeting of $v_{i,j,n}$ and $v_{i,j,m}$, then either $\mathcal{V}^k(H_1, S) = \emptyset$ or $\mathcal{V}^k(H_2, S) = \emptyset$, but not both.*

Proof. Since $v_{i,j,m}$ is new, $\mathcal{D}_m \notin H_1 \bigcup H_2$. Therefore, the maximum circumference C_1 centered at $v_{i,j,m}$ contains $\mathcal{D}_n$ but that, C_2, centered at $v_{i,j,n}$ does not contain $\mathcal{D}_m$. W.l.o.g, we sketch the relative locations of $\mathcal{D}_i$, $\mathcal{D}_j$, $\mathcal{D}_n$, and $\mathcal{D}_m$ as shown in Figure 2.16. Let p_i, p_j, and p_m be the tangent points of $\mathcal{D}_i$, $\mathcal{D}_j$, and $\mathcal{D}_m$ with C_1, respectively. As the event occurs, $C_1 \equiv C_2$. Thus, $\mathcal{D}_n$ must first touch C_1 at arc $p_m p_j$, or $p_i p_m$. If it touches C_1 at arc $p_m p_j$, any disk that contains $\mathcal{D}_i$ and $\mathcal{D}_n$ must touch either $\mathcal{D}_m$ or $\mathcal{D}_j$. Therefore, $\mathcal{V}^k(H_1, S) = \emptyset$. When $C_1 \equiv C_2$, C_1 (and C_2) contains $H_2 \backslash \{\mathcal{D}_n, \mathcal{D}_j\}$. Therefore, as $\mathcal{D}_n$ expands further a sufficiently small amount, there

exists a disk internally tangent to $\mathcal{D}_n$, $\mathcal{D}_j$, and $\mathcal{D}_m$, and contains $H_2 \backslash \{\mathcal{D}_i\}$. Therefore, $\mathcal{V}^k(H_2, S) \neq \emptyset$. Similar arguments apply as $\mathcal{D}_n$ first touches C_1 at $p_i p_m$, which gives rise to the case $\mathcal{V}^k(H_2, S) = \emptyset$ and $\mathcal{V}^k(H_1, S) \neq \emptyset$. This completes the proof. □

Next, we take a closer look to how a face emerges and disappears. We characterize the structure of a region immediately before it disappears. In addition, we describe the structure of the region that emerges as the result of another region's death.

Lemma 2.8. *Consider an edge $e_{i,j}$ with an old vertex $v_{i,j,n}$ and a new vertex $v_{i,j,m}$ of 2 regions $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$, where $\{\mathcal{D}_i, \mathcal{D}_n\} \subset H_1$ and $\{\mathcal{D}_j, \mathcal{D}_n\} \subset H_2$. Assume that when $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}(o_n, r_{n'})$, the two vertices of $e_{i,j}$ meet resulting in the degeneration of $\mathcal{V}^k(H_1, S)$, then there exists an positive ϵ such that when $\mathcal{D}_n$ expands to $\mathcal{D}_{n''}(o_n, r_{n'} - \epsilon)$, $\mathcal{V}^k(H_1, S) \neq \emptyset$ and $\mathcal{V}^k(H_1, S)$ contains only 4 vertices, including 2 old vertices, namely, $v_{i,m,n}$ and $v_{i,j,n}$, and 2 new vertices, namely, $v_{i,j,m}$ and $v_{j,n,m}$.*

Proof. Let C be the maximum circumference centered at $v_{i,j,m}$. Let p_i, p_j, and p_m be the tangent points of C_1 with $\mathcal{D}_i$, $\mathcal{D}_j$, and $\mathcal{D}_m$, respectively. When $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}(o_n, r_{n'})$, $\mathcal{V}^k(H_1, S) = \emptyset$. Thus, $\mathcal{D}_n$ is internally tangent with C at arc $p_m p_j$, which does not contains p_i. When $\mathcal{D}_n$ expands to $\mathcal{D}_{n''}(o_n, r_{n'} - \epsilon)$, obviously, $v_{i,j,m}$ remains a new vertex, and $v_{i,j,n}$ is an old vertex of $\mathcal{V}^k(H_1, S)$. In addition, there exists a circle that is internally tangent to $\mathcal{D}_n, \mathcal{D}_m, \mathcal{D}_j$, and contains $\mathcal{D}_i$ and all disks in $H_1 \backslash \{\mathcal{D}_i, \mathcal{D}_n\}$. Thus, $v_{j,n,m}$ is a new vertex of $\mathcal{V}^k(H_1, S)$. Meanwhile, there exists a circle that is internally tangent to $\mathcal{D}_{n'}, \mathcal{D}_m, \mathcal{D}_i$, and all disks in $H_1 \backslash \{\mathcal{D}_i, \mathcal{D}_n\}$. Therefore, $v_{i,m,n}$ is an old vertex of $\mathcal{V}^k(H_1, S)$. Now we show that there exists no

other vertices of $\mathcal{V}^k(H_1, S)$. In fact, as $\mathcal{D}_n$ expand to $\mathcal{D}_{n'}$ to touch C, any circle C' that is tangent to any disk $\mathcal{D}_h \in H_1$ and contains $H_1 \backslash \{\mathcal{D}_h\}$ must contains either $\mathcal{D}_j$ or $\mathcal{D}_m$, or both. Obviously, there exists an $\epsilon > 0$ such that when $\mathcal{D}_n$ shrinks ϵ from $\mathcal{D}_{n'}$, such property of C' still holds. This completes the proof. □

Lemma 2.9. *Assume that the new vertex $v_{i,j,n}$ meets the old one $v_{i,j,m}$ as $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}(o_n, r_{n'})$, which results in the degeneration of face $\mathcal{D}^k(H_1, S)$, where $H_1 = H \bigcup \{\mathcal{D}_n, \mathcal{D}_i\}$. Prior to the degeneration, $\mathcal{D}^k(H_1, S)$ shares edges $e_{j,n}$, $e_{m,n}$, $e_{m,i}$, and $e_{j,i}$ with faces $\mathcal{V}^k(H_2, S)$, $\mathcal{V}^k(H_3, S)$, $\mathcal{V}^k(H_4, S)$, and $\mathcal{V}^k(H_5, S)$, respectively, where $H_2 = H \bigcup \{\mathcal{D}_i, \mathcal{D}_j\}$, $H_3 = H \bigcup \{\mathcal{D}_i, \mathcal{D}_m\}$, $H_4 = H \bigcup \{\mathcal{D}_n, \mathcal{D}_m\}$, $H_5 = H \bigcup \{\mathcal{D}_j, \mathcal{D}_n\}$. There exists $\epsilon > 0$ such as the following holds: i) $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S) \neq \emptyset$, and ii) edges $e_{j,n}$, $e_{m,n}$, $e_{m,i}$, and $e_{j,i}$ of regions $\mathcal{V}^k(H_2, S)$, $\mathcal{V}^k(H_3, S)$, $\mathcal{V}^k(H_4, S)$, $\mathcal{V}^k(H_5, S)$ are replaced by edges $e_{i,m}$, $e_{i,j}$, $e_{n,j}$, $e_{n,m}$ with region $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S)$, respectively. Thus, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S)$ consists of 4 edges, namely, $e_{i,m}$, $e_{i,j}$, $e_{n,j}$, and $e_{n,m}$, and 4 vertices, including 2 new vertices, namely, $v_{i,n,m}$ and $v_{i,j,n}$, and 2 old vertices, namely, $v_{i,j,m}$, and $v_{j,n,m}$.*

Proof. As shown in Figure 2.16, when $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}(o_n, r_{n'})$, C_1 is internally tangent to $\{\mathcal{D}_i, \mathcal{D}_j, \mathcal{D}_n, \mathcal{D}_m\}$ and contains H. Clearly, C_1 does not contains $\mathcal{D}_n$ as it expands further from $\mathcal{D}_{n'}$. Since $|H| = k - 2$, $v_{i,j,m}$ is an old vertex of region $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S)$ due to Proposition 2.5. This proves claim *i*). To show claim *ii*), we only show the case $e_{j,n}$ being replaced by $e_{i,m}$. The other claims can be proved similarly. As $\mathcal{D}_n := \mathcal{D}_{n'}(o_n, r_{n'} + \epsilon)$, the circle C that is internally tangent to $\mathcal{D}_n$ and $\mathcal{D}_j$ and contains $\mathcal{D}_i$ must contain $\mathcal{D}_m$. Since $\mathcal{D}_m \notin H_2 \bigcup H_1$, $e_{j,n}$ is not an edge of $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$ according to Proposition 2.5. Also, since $v_{i,j,m}$ is

a valid vertex, there exists an edge $e_{i,m}$ of $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_j\})$ and $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\})$. Let $H_5 = H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\})$. There exists a small ϵ such that the circle containing $H_5 \backslash \{\mathcal{D}_h\}$ and is internally tangent to $\mathcal{D}_h \in H_5$ and some $\mathcal{D}_q \notin H_5 \cup \{\mathcal{D}_j, \mathcal{D}_m\}$ must contains $\mathcal{D}_n$ or $\mathcal{D}_i$. Thus, there are no other edges of $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\})$. This completes the proof. □

Lemma 2.7 claims that as a new vertex meets an old one due to a disk's expansion, one of the two faces that is incident to the edge connecting the two vertices degenerates to a point and eventually disappears. Lemma 2.8 describes the structure of the face before it disappears, as illustrated in Figure 2.17b. Lemma 2.9 characterizes the structure of the newly-born face (Figure 2.17d). The transform of the diagram in this case, as a face disappears giving birth to another face, is intuitive when viewed from the structure of the corresponding order-$(k-1)$ max VD. In essence, the expansion of $\mathcal{D}_n$ leads to the degeneration of the edge $e_{i,n}$ of faces $\mathcal{V}^{k-1}(H \bigcup \{\mathcal{D}_i\}, S)$ and $\mathcal{V}^{k-1}(H \bigcup \{\mathcal{D}_n\}, S)$, and simultaneously, the birth of edges $e_{j,m}$ of faces $\mathcal{V}^{k-1}(H \bigcup \{\mathcal{D}_j\}, S)$ and $\mathcal{V}^{k-1}(H \bigcup \{\mathcal{D}_m\}, S)$, as illustrated in Figures 2.17a and 2.17c. To this end, we conclude the discussion in the chain of changes in the order-k max VD due the expansion of a type-I disk. We now move on to the analysis of type-II disks' expansion.

2.3.3 Expansion of a type-II disk

Let $\mathcal{D}_i$ be a type-II disk, i.e., $e_{i,j} = \emptyset$, $\forall \mathcal{D}_j \in S, j \neq i$. We first make some claims regarding to general type-II disks.

Proposition 2.6. *$\mathcal{D}_i$ is a type-II disk in $V^k(S)$ iff $\mathcal{D}_i \in H$ for all H's such that $\mathcal{V}^k(H,S) \neq \emptyset$, or $\mathcal{D}_i$ contains k disks.*

Proof. If $\mathcal{D}_i \in H$ for all H's such that $\mathcal{V}^k(H,S) \neq \emptyset$, then clearly $e_{i,j} = \emptyset, \forall \mathcal{D}_j \in S$. If $\mathcal{D}_i$ contains k disks $\mathcal{D}_1, \ldots, \mathcal{D}_k$, then $d_{\max}(p, \mathcal{D}_i) > d_{\max}(p, \mathcal{D}_j) \forall p$, where $j \in \{1, \ldots, k\}$. Therefore, $\mathcal{V}^k(H,S) = \emptyset$, where $H \ni \mathcal{D}_i$, thus, $e_{i,j} = \emptyset, \forall \mathcal{D}_j \in S$. This proves the sufficient condition. To show the necessary condition, we consider 2 cases: $\exists H \ni \mathcal{D}_i$ such that $\mathcal{V}^k(H,S) \neq \emptyset$, or $\nexists H \ni \mathcal{D}_i$ such that $\mathcal{V}^k(H,S) \neq \emptyset$. We prove by contradiction. In the former case, we show that $\mathcal{D}_i \in H$ for all H's such that $\mathcal{V}^k(H,S) \neq \emptyset$. Assume there exist 2 regions, e.g., $\mathcal{V}^k(H_1,S)$ and $\mathcal{V}^k(H_2,S)$ such that $\mathcal{D}_i \in H_1$ and $\mathcal{D}_i \notin H_2$. This implies that $e_{i,j} \neq \emptyset$ according to [43], a contradiction. In the latter case, we show that $\mathcal{D}_i$ contains k disks. Assume $\mathcal{D}_i$ contains l disks $(0 \leq l < k)$. W.l.o.g, let $d_{\max}(o_i, \mathcal{D}_1) \leq \ldots \leq d_{\max}(o_i, \mathcal{D}_l) < d_{\max}(o_i, \mathcal{D}_i)$. Since $\mathcal{D}_i$ does not contain other disks, let $d_{\max}(o_i, \mathcal{D}_{l+1}) \leq \ldots \leq d_{\max}(o_i, \mathcal{D}_{k-1}) \leq \ldots \leq d_{\max}(o_i, \mathcal{D}_{|S|})$. Therefore, $o_i \in \mathcal{V}^k(\{\mathcal{D}_i, \mathcal{D}_1, \ldots, \mathcal{D}_{k-1}\}, S)$. Thus, $\mathcal{V}^k(H,S) \neq \emptyset$, where $H = \{\mathcal{D}_i, \mathcal{D}_1, \ldots, \mathcal{D}_k - 1\}$, a contradiction. This completes the proof. □

Proposition 2.7. *Consider a type-II disk $\mathcal{D}_i$ in $V^k(S)$, where $\mathcal{D}_i$ does not contain k other disks. Expanding any disks $\mathcal{D}_j \in S$ $(j \neq i)$ does not make $\mathcal{D}_i$ type-I.*

Proof. Let $\mathcal{D}_i$ be a type-II disk, and p is an arbitrary point in the plane. Since $\mathcal{D}_i \in H$ for all H's such that $\mathcal{V}^k(H,S) \neq \emptyset$, there exist $k-1$ disk $d_{\max}(p, \mathcal{D}_1) \leq d_{\max}(p, \mathcal{D}_2) \leq \ldots \leq d_{\max}(p, \mathcal{D}_j - 1) < d_{\max}(p, \mathcal{D}_i) < d_{\max}(p, \mathcal{D}_j) \leq \ldots \leq d_{\max}(p, \mathcal{D}_{k-1})$, i.e., the disk C centered at any point p and containing k disks must contain $\mathcal{D}_i$ (C is not internally tangent to $\mathcal{D}_i$). This remains to be true with the expansion of any disk

other than $\mathcal{D}_i$. This completes the proof $\square$

As a result, a type-II disk $\mathcal{D}_i$ must be in the interior of all maximum circumferences of $V^k(S)$. Therefore, when $\mathcal{D}_i$ expands, it must first touch a maximum circumference associated with some edge $e_{n,m}$. However, we claim that it can only touch a maximum circumference centered at a (possibly infinite) vertex of $e_{n,m}$. Assume $\mathcal{D}_i$ first touches the maximum circumference C associated with $\mathcal{D}_n$ and $\mathcal{D}_m$ centered at O as shown, w.l.o.g., in Figure 2.18, i.e., C contains $k-1$ disks, H. We can move O along $b_{n,m}$ further away from $\mathcal{D}_i$ to O' such that C', the maximum circumference centered at O', still contains H. Since all maximum circumferences contain type-II disks, there exists ϵ such that C' is internally tangent to $\mathcal{D}_i(o_i, r_i - \epsilon)$. As O is moving along $b_{n,m}$, C will eventually be internally tangent to some disk of S, which makes C the maximum circumference centered at a vertex of $V^k(S)$. This shows that $\mathcal{D}_i$ must first touch a maximum circumference centered at a vertex as it expands. Furthermore, we observe that such an expanding disk must be first internally tangent to a maximum circumference centered at an infinite vertex. Before delving into the detailed analysis, we sketch an example in Figure 2.19, which shows the order-4 Voronoi diagram of 5 disks of zero radius, $S = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_5\}$. The Voronoi regions are $\mathcal{V}^4(\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_4, \mathcal{D}_3\}, S)$, $\mathcal{V}^4(\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_5, \mathcal{D}_3\}, S)$, $\mathcal{V}^4(\{\mathcal{D}_1, \mathcal{D}_5, \mathcal{D}_4, \mathcal{D}_3\}, S)$, and $\mathcal{V}^4(\{\mathcal{D}_2, \mathcal{D}_4, \mathcal{D}_5, \mathcal{D}_3\}, S)$, which make $\mathcal{D}_3$ a type-II disk. The diagram (dashed) consists of 2 finite vertices, v_1 and v_2, whose corresponding maximum circumferences are C_6, and C_7 shown by lighter circles. The maximum circumferences centered at the infinite end of edges e_i are shown by straight lines C_i, $i \in \{1, 2, 3, 4\}$. As shown

in the figure, as $\mathcal{D}_3$ expands (dashed circle), it first touches C_5, the maximum circumference centered at the infinite end of edge e_5 at a point in segment $\mathcal{D}_2\mathcal{D}_5$. We formalize this in the following lemma.

Lemma 2.10. *Consider a disk $\mathcal{D}_i$, which is not in the interior of any disk, and shares no edge with other disks. Upon expansion, $\mathcal{D}_i$ must first touch a maximum circumference C centered at the infinite vertex of some edge. Let the edge be $e_{n,m}$, and the tangent points of $\mathcal{D}_n$ and $\mathcal{D}_m$ with C be p_m and p_n. Then, $\mathcal{D}_i$ is tangent to C on the segment $p_m p_n$.*

Proof. We prove the claims by contradiction. Let $\mathcal{D}_{i'}$ be the expanded $\mathcal{D}_i$. W.l.o.g, assume $\mathcal{D}_{i'}$ touches the maximum circumference C centered at $v_{n,m,k}$. Let p_n, p_m, and p_k be the tangent points of $\mathcal{D}_n, \mathcal{D}_m$, and $\mathcal{D}_k$ with C, respectively. Assume $\mathcal{D}_{i'}$ is tangent to C at a point in arc $p_k p_m$. We consider 2 cases: $v_{n,m,k}$ is old or new. When $v_{n,m,k}$ is new, there exists $H \subset S$, $|H| = k-1$, $\mathcal{D}_i \in H$, such as $\mathcal{V}^k(\{\mathcal{D}_n\} \bigcup H, S) \neq \emptyset$, $\mathcal{V}^k(\{\mathcal{D}_m\} \bigcup H, S) \neq \emptyset$, and $\mathcal{V}^k(\{\mathcal{D}_k\} \bigcup H, S) \neq \emptyset$. As $\mathcal{D}_{i'}$ expands further to $\mathcal{D}_{i''}$, any circle that contains $\mathcal{D}_n$ and $\mathcal{D}_{i''}$ must contain either $\mathcal{D}_k$, or $\mathcal{D}_m$. Therefore, $\mathcal{V}^k(\{\mathcal{D}_n\} \bigcup H, S) = \emptyset$ in the updated diagram. Meanwhile, since $\mathcal{D}_{i'}$ is first tangent to C, there exists a maximum circumference C' centered at another vertex of $\mathcal{V}^k(\{\mathcal{D}_n\} \bigcup H, S)$ that contains H, which implies that $\mathcal{V}^k(\{\mathcal{D}_n\} \bigcup H, S) \neq \emptyset$ in the updated diagram, a contradiction. If $v_{n,m,k}$ is old, there exists $H \subset S$, $|H| = k-2$, $\mathcal{D}_i \in H$, such as $\mathcal{V}^k(\{\mathcal{D}_n, \mathcal{D}_m\} \bigcup H, S) \neq \emptyset$, $\mathcal{V}^k(\{\mathcal{D}_m, \mathcal{D}_k\} \bigcup H, S) \neq \emptyset$, and $\mathcal{V}^k(\{\mathcal{D}_k, \mathcal{D}_n\} \bigcup H, S) \neq \emptyset$. We first show that as $\mathcal{D}_{i'}$ expands further to $\mathcal{D}_{i''}$, either $\mathcal{V}^k(\{\mathcal{D}_n, \mathcal{D}_m\} \bigcup H, S) = \emptyset$ or $\mathcal{V}^k(\{\mathcal{D}_k, \mathcal{D}_n\} \bigcup H, S) = \emptyset$ in the updated diagram by contradiction. Assume both faces exist. Since they differ by $\mathcal{D}_k$ and $\mathcal{D}_m$, there

exists an edge $e_{k,m}$ of 2 faces. Meanwhile, recall that $\mathcal{D}_{i'}$ is tangent to C at a point in arc $p_k p_m$, there is no circle internally tangent to $\mathcal{D}_k$ and $\mathcal{D}_m$, that contains both $\mathcal{D}_n$ and $\mathcal{D}_{i''}$. Thus $e_{k,m} = \emptyset$, a contradiction. Therefore, either face does not exist. W.l.o.g, assume $\mathcal{V}^k(\{\mathcal{D}_n, \mathcal{D}_m\} \bigcup H, S) = \emptyset$. Since $\mathcal{D}_{i'}$ is first tangent to C, the other vertex, e.g., $v_{j,m,k}$, of edge $e_{k,m}$ exists. Thus, there is a maximum circumference centered at $v_{j,m,k}$ which contains $\{\mathcal{D}_n, \mathcal{D}_{i''}\} \bigcup H$. This implies that $\mathcal{V}^k(\{\mathcal{D}_n, \mathcal{D}_m\} \bigcup H, S) \neq \emptyset$, a contradiction. This completes the first claim. Next, let C be the maximum circumference centered at the infinite end of edge $e_{n,m}$ of $\mathcal{V}^k(H, S)$, where $\mathcal{D}_n \in H$. Let p_n and p_m be the tangent points of C with $\mathcal{D}_n$ and $\mathcal{D}_m$. Since C is infinitely large, arc $p_n p_m$ is a line segment. Let p be the tangent point of $\mathcal{D}_i$ with C. Assume p lies outside $p_n p_m$ in the ray that does not contain p_n. Clearly, C contains $H \backslash \{\mathcal{D}_i, \mathcal{D}_n\}$. Now, we can always shrink $\mathcal{D}_i$ by a sufficiently small ϵ such that there exists a line d tangent to $\mathcal{D}_i(o_i, r_i - \epsilon)$ and $\mathcal{D}_m$ such that a half-plane created by d contains H. This means that $e_{n,i} \neq \emptyset$, a contradiction. Similarly, we can show that $\mathcal{D}_i$ is not exterior to $p_n p_m$ in the part that does not contain p_m. This completes the proof. □

We now study further what happens posterior to the contact. As $\mathcal{D}_i$ is internally tangent to C, a new face is born, which is due to two infinite edges and one old vertex. Formally,

Lemma 2.11. *Let $e_{n,m}$ be an infinite edge shared by faces $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$, and $v_{k,n,m}$ be a vertex of $e_{n,m}$ in $V^k(S)$. Let C be the maximum circumference associated with $e_{n,m}$ and centered at infinity that is internally tangent with $\mathcal{D}_n$ and $\mathcal{D}_m$ at p_n and p_m, respectively. Let $\mathcal{D}_i$ be a type-II disk. Assume $\mathcal{D}_i$ expands and first*

touches C at a point in arc p_np_m. Then, there exists further expansion $\epsilon > 0$ of $\mathcal{D}_i$ such that : i) $\mathcal{V}^k((H_1 \bigcup H_2)/\mathcal{D}_i, S) \neq \emptyset$, ii) $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$ share an edge with $\mathcal{V}^k((H_1 \bigcup H_2)/\mathcal{D}_i, S)$, i.e., $e_{i,n}, e_{i,m}$, respectively, and iii) $e_{i,n}$ and $e_{i,m}$ are infinite edges.

Proof. To prove the first two claims, it is sufficient to show that there is a vertex of faces $\mathcal{V}^k((H_1 \bigcup H_2)/\mathcal{D}_i, S)$, $\mathcal{V}^k(H_1, S)$ and $\mathcal{V}^k(H_2, S)$. In fact, as $\mathcal{D}_i$ touches C at a point in arc p_np_m, there exists an $\epsilon > 0$ such that there is a circle that is internally tangent to $\mathcal{D}_n, \mathcal{D}_m$, and $\mathcal{D}_{i'}(o_i, r_i + \epsilon)$, and contains the set $(H_1 \bigcup H_2)/\{\mathcal{D}_{i'}, \mathcal{D}_n, \mathcal{D}_m\}$, whose cardinality is $k-2$. Thus, $v_{i',n,m}$ is an old vertex according to Proposition 2.5. This implies that $\mathcal{V}^k((H_1 \bigcup H_2)/\mathcal{D}_i, S) \neq \emptyset$. To see that $e_{i,n}$ is infinite, it suffices to show that there is a straight line, d, that is tangent to $\mathcal{D}_n$ and $\mathcal{D}_i$, and divides the plane into 2 halves, one of which contains $H_1 \bigcup H_2$. Since C is infinitely large, there is a half-plane divided by C that contains $H_1 \bigcup H_2$. Since $\mathcal{D}_i$ first touch p_np_m as it expands, we can always find a small expansion ϵ of $\mathcal{D}_i$ such that there exists a line d that is tangent to $\mathcal{D}_n$ and $\mathcal{D}_i$, and divides the plane into 2 halves, one of which contains $H_1 \bigcup H_2$. (In fact, d is very close to C.) This completes the proof. □

Lemma 2.11 describes the max VD structure due to an event caused by a type-II disk's expansion. Lemmas 2.10 characterizes the expansion that leads to such an event by keeping track of infinite edges. However, since the total number of infinite edges is $O(N)$, the computational complexity of identifying such events remains high. Fortunately, it can be shown that transformation of any type-I to type-II disk in a max VD $V^k(S)$ that does not contains type-II disks is possible only when $k = N$.

Moreover, it takes $O(k^2N)$ for all type-II disks of $V^k(S)$ to transform to type-I disks. We formalize the claims in the following lemma.

Lemma 2.12. *Assume there is no type-II disk in $V^k(S)$. $\mathcal{D}_n$ expands but it does not contain k disks. If any disk $\mathcal{D}_i$, $i \neq n$, becomes type-II due to the expansion of $\mathcal{D}_n$, then $k = N$.*

Proof. When a disk expands, two vertices meet and the associated edge degenerates. Depending on the type of vertices, three cases may arise. First, both vertices are new. Let the associated edge be $e_{n,m}$, and it is at intersection of two faces as shown in Figure 2.21a. If either vertex, e.g., $v_{n,m,j}$ is finite, an edge $e_{j,m}$ appears as edge $e_{n,m}$ degenerates. Therefore, there exist edges of disks $\mathcal{D}_n, \mathcal{D}_m, \mathcal{D}_i$, and $\mathcal{D}_j$. If both vertices of $e_{n,m}$ are infinite, the maximum circumferences centered at these two infinite vertices are two lines n_1m_1 and n_2m_2, that are outer tangent to the two disks, as illustrated in Figure 2.20b. As point p traverses on $e_{n,m}$ from one end to another, the tangent point of the maximum circumference centered at p with $\mathcal{D}_m$ moves along the arc a_1 from m_1 to m_2 (or vice versa). Since any maximum circumference associated with $e_{n,m}$ contains H. Thus, H must be in the plane limited by two lines n_1m_1 and n_2m_2, and two arcs a_1 and a_2. If $e_{n,m}$ disappears, $\mathcal{D}_n$ must contains $\mathcal{D}_m$, and thus, it must contains H. Since $\mathcal{D}_n$ does not contain k disks, $e_{n,m}$ remains. Therefore, no type-I disk becomes type-II as a result of the meeting of two new vertices.

Meeting of a new vertex with an old one shown in Figure 2.21b clearly does not make any disk type-II. In the case of meeting of 2 old vertices (Figure 2.21c), e.g., v_1 and v_2, edge $e_{i,j}$ degenerates. If $\mathcal{D}_j \equiv \mathcal{D}_\infty$ and $e_{i,j}$ is the only edge associated with $\mathcal{D}_j$

prior to the expansion, $\mathcal{D}_j$ becomes type-II. Next, we show that this only happens when $k = N$. Let $H'' = H \backslash \{\mathcal{D}_i, \mathcal{D}_n, \mathcal{D}_m\}$. Let $\mathcal{D}_{n'}(o_n, r_{n'})$ be the expanded $\mathcal{D}_n$ such that 2 vertices in Figure 2.21c meet. As $\mathcal{D}_j = \mathcal{D}_\infty$, there exists a tangent line l_1 of $\mathcal{D}_n$ and $\mathcal{D}_i$ such that one of the half-planes created by l_1 contains H (Figure 2.20a). Similarly, there exists a tangent line l_2 such that one of the half-planes created by l_2 contains H. As shown in Section 2.3.1, there exists infinite edges, i.e., $e_{n,i}$ of 2 faces $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_i, \mathcal{D}_m\})$ and $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_n, \mathcal{D}_m\})$ and $e_{m,i}$ of 2 faces $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_n, \mathcal{D}_m\})$ and $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_n, \mathcal{D}_i\})$. As $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}$, l_1 rotates as illustrated in the figure. When v_1 meets v_2, lines l_1 and l_2 coincide and both are outer tangent to $\mathcal{D}_{n'}$, $\mathcal{D}_i$, and $\mathcal{D}_m$. Thus, we can always shrink $\mathcal{D}_{n'}$ by a small amount ϵ such that the line tangent to $\mathcal{D}_{n'}(o_n, r_{n'} - \epsilon)$ and $\mathcal{D}_m$ creates a half-plane that contains $\mathcal{D}_{n'}$, $\mathcal{D}_m$ and $H'' \backslash \{\mathcal{D}_i\}$. This implies that there exists an infinite edge $e_{n,m}$ of 2 faces $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_i, \mathcal{D}_n\})$ and $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_i, \mathcal{D}_m\})$. Moreover, there exists a disk internally tangent to $\mathcal{D}_{n'}$, $\mathcal{D}_m$, and $\mathcal{D}_i$ that contains H''. Clearly, there does not exists edge of $\mathcal{D}_h \in H''$, $\mathcal{D}_h \notin \{\mathcal{D}_m, \mathcal{D}_i, \mathcal{D}_{n'}\}$. This implies that as $\mathcal{D}_n$ expands to $\mathcal{D}_{n'}(o_n, r_{n'} - \epsilon)$, $V^{k-1}(S)$ consists of 3 faces, i.e., $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_i, \mathcal{D}_m\})$, $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_i, \mathcal{D}_n\})$, and $\mathcal{V}^{k-1}(H'' \bigcup \{\mathcal{D}_n, \mathcal{D}_m\})$. This suffices to show that $|H''| = N - 3$. Thus $k = N$. □

The lemma indicates that, in a general order-k max VD of N disks with no type-II disk, where $k < N$, the expansion of disks does not create any type-II disk. We next show that it takes $O(k^2 N)$ to pre-process an order-k VD such that all disks are type-I. The idea is to expand type-II disks until they are no longer type-II. We further constrain that the expanding disk must not contain any disks. We first observe that

an expanding type-II disk $\mathcal{D}_n$ does not contain any type-I disk $\mathcal{D}_i$. Otherwise, let $e_{i,j}$ be an edge of $V^k(S)$, then any maximum circumference associated with $e_{i,j}$ does not contain $\mathcal{D}_n$. Since $\mathcal{D}_i$ is type-II, there exists $H \in S$ such that $\mathcal{V}^k(H, S) \neq \emptyset$. This implies that $\mathcal{D}_n \notin H$, a contradiction by Proposition 2.6. Next, we show the procedure to transform all type-II disks in $V^k(S)$ to type-I in $O(k^2n)$ time. Let S'' be the set of type-II disks of $V^k(S)$ ($|H''| = h$). Let d_i be the minimum size of a type-II disk $\mathcal{D}_i \in S''$, such that it is tangent to some maximum circumference centered at an infinite vertex. We sort d_i's in the ascending order, i.e., $d_1 \leq d_2 \leq ... \leq d_h$. We sequentially update $V^k(S)$ by expanding $\mathcal{D}_i$, $i = 1 \ldots h$, and set the radii of disks $\mathcal{D}_m$, $i < m \leq h$, to be r_i, i.e., the radius of disk $\mathcal{D}_i$. We claim that d_m is still the minimum size that $\mathcal{D}_m$ needs to expand to become type-I after the expansion of $\mathcal{D}_i$, $i = 1, \ldots, m-1$, to their respective radii. Assume the expansion of $\mathcal{D}_m$ first touches the maximum circumference centered at the infinite vertex on $e_{i,j}$ prior to the expansions of $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{m-1}$. Let p_i and p_j be the tangent points of the maximum circumference with $\mathcal{D}_i$ and $\mathcal{D}_j$, respectively. If $\mathcal{D}_m$ is the only type-II disk tangent to p_ip_j, then the changes in $V^k(S)$ is given in Lemma 2.11. Otherwise, if there is another type-II disk, say $\mathcal{D}_n$ ($n < m$) tangent to p_ip_j at p_n. When such an event occurs, one end of $e_{i,j}$ becomes finite, and two infinite edges, i.e., $e_{i,n}$ and $e_{j,n}$ are born. Thus, we need to compute the minimum size of $\mathcal{D}_m$ when it touches the maximum circumferences centered at two infinite ends of $e_{i,n}$ and $e_{j,n}$. In the other words, we compute the size of $\mathcal{D}_m$ so that it touches p_ip_n or p_np_j. Since p_n is on p_ip_j, the amount of expansion for $\mathcal{D}_n$ to touch them is equal to that for it to touch p_ip_j. This proves the above claim. In general, determining the position in the segment p_ip_j

that $\mathcal{D}_n$ touches can be done by a search among the sub-segments of p_ip_j. Thus, the cost of updating $V^k(S)$ due to the expansion of type-II disk $\mathcal{D}_m$ is $O(h' \log h')$, where h' is the number of type-II disks touching p_ip_j first. The total cost of expanding h type-II disks is $\sum_{h'}(h' \log h') = O(h \log h)$. The minimum expansion for a type-II disk to become type-I is computed by considering all infinite vertices, which takes $O(kN)$. Thus, the total cost of transforming h type-II disks into type-I disks is $O(hkN)$. Since $h \leq k$, the worst-case time complexity is $O(k^2N)$. We summarize the procedure in Algorithm 2.1. Line 9 indicates that if a type-II disk's given size is smaller than the size that makes it type-I, then it is type-II irrespective to any changes of other disks due to Proposition 2.7.

We now prove Theorem 2.1.

2.3.3.0.1 Proof of Theorem 2.1 It is known from [43] that the number of edges and faces in the order-k Voronoi diagram of disk centers, including pseudo edges, is $O(kN)$. From the above results, when disks expand, the total number of edges and faces in the order-k max VD does not exceed $O(kN)$. Since there are $O(kN)$ faces and N disks, the expected number of faces corresponding to a disks is k, each has 6 edges on average [7]. In addition, since disks either expand to infinite size, or do not contain other disks, there are no events due to the exclusion of $\mathcal{D}_i$ from other disks when it expands. Thus, when a disk $\mathcal{D}_i$ expands, the maximum number of events is equal to the number of edges in faces corresponding to $\mathcal{D}_i$. Therefore, the expected events due to $\mathcal{D}_i$'s expansion is $O(k)$. Therefore, the expected number of events corresponding to N disks' expansions is $O(kN)$. This completes the proof.

Algorithm 2.1: Pre-process order-k Maximum Voronoi diagrams of disks

input : $V^k(S)$

output: $V^k(S)$ that does not contains type-II disks

```
1 S'' ← type-II disks in S ;
2 foreach D_i ∈ S'' do
3     d[i] ← minimum expansion of D_i to be first tangent to a maximum
      circumference ;
4 sort S'' in ascending order based on d;
5 foreach D_i ∈ S'' do
6     if d[i] ≤ targeted size of D_i then
7         modify V^k(S) according to Lemma 2.11;
8     else
9         assign the targeted radius to D_i ;
```

We are now in the position to sketch algorithms for constructing order-k maximum Voronoi diagrams of disks and order-k Voronoi diagrams of points in 2D based on the principles of expanding/shrinking disks.

2.4 Application of disk expansion/shrinking to order-k max VDs

We show that properties of disk expansion/shrinking in order-k max VDs can be applied to order-k max VD construction, insertion and deletion of sites in order-k max VDs which leads to order-k VD construction.

2.4.1 Construction of order-k max Voronoi diagrams of disks in 2D

In constructing the order-k max VD of disks S, we start with an order-k Voronoi diagram (VD) of disk centers and iteratively expand each disk in S by a fixed amount $d_{\min}$, where $d_{\min} \stackrel{\Delta}{=} \min_{i,j \in S} d(o_i, o_j) - \epsilon$. We stop when all disks reach their targeted size. The resulted diagram is the order-k max VD of S. Let $r_{\max} = \max_{i \in S} r_i$ and $r_{\min} = \min_{i \in S} r_i$. Clearly, the total number of rounds a disk needs to expand is bounded by $\lceil \frac{r_{\max} - r_{\min}}{d_{\min}} \rceil$. This implies that the algorithm terminates. Since disks expand by $d_{\min}$, they are not contained in other disks until they reach their targeted sizes. Furthermore, when a disk contains k other disks in its expansion, it becomes type-II since the k disks have reached their targeted sizes. Thus, it further expansion

does not change the diagram. Therefore, the algorithm proceeds in such a way that all expanding disks are always type-I. As discussed in the previous sections, expanding disks does not make any new type-II disk. Thus, it is always possible to evaluate the expansion such that the next vertex meeting happens. The procedure of order-k max VD construction is summarized in Algorithm 2.2.

We now analyze the running time of the algorithm. Algorithm 2.2 takes the set S of N disks as inputs. It starts by constructing the order-k VD of disk centers (line 2), which in face is the order-k max VD of disks whose radii are all equal to the minimum radius of N disks, denoted as S'. The process costs $O(k^2 N \log N)$ in running time ([43]). Since the order-k VD may contain type-II disks, we first make them type-I by executing Algorithm 2.1 (lines 4 - 4). This takes $O(k^2 N)$. Then, we iteratively scan all disks in S and expand those whose radii are smaller than their respective sizes an amount $d_{\min}$ (lines 6 - 17). $d_{\min}$ can be computed in $O(N)$ using the order-1 VD of S', which is a byproduct in the incremental construction of $V^k(S')$. Line 11 requires a sorted list of expansions. When an event happens, an expansion is inserted into the sorted list. In Algorithm 2.2, each disk expands at most $\lceil \frac{r_{\max} - r_{\min}}{d_{\min}} \rceil$ times. Since the expected number of events when each disk expands once is $O(kN)$, the expected cost of lines 6 - 17 is $O(\lceil \frac{r_{\max} - r_{\min}}{d_{\min}} \rceil kN \log N)$. Therefore,

Theorem 2.2. *The order-k max VD of N disks can be constructed in*

$$O\left(k^2 N \log N + \left\lceil \frac{r_{\max} - r_{\min}}{d_{\min}} \right\rceil kN \log N\right)$$

expected time, where $r_{\max}$ and $r_{\min}$ are respectively the maximum and minimum radii of N disks, and $d_{\min}$ is the minimum distance between 2 disks' centers.

Algorithm 2.2: Order-k Maximum Voronoi diagram of disks

input : A set of disks $S = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_1(o_2, r_2), \ldots, \mathcal{D}_N(o_N, r_N)\}$

output: The order-k max VD of S, $V^k(S)$

$r_{\min} \leftarrow \min_i r_i$;

$S' \leftarrow \{\mathcal{D}_1(o_1, r'_1 = r_{\min}), \ldots, \mathcal{D}_N(o_N, r'_N = r_{\min})\}$;

Construct $V^k(S')$;

if $V^k(S')$ *contains type-II disks* **then** pre-process $V^k(S')$ using Algorithm 2.1;

$d_{\min} \leftarrow \min_{i,j \in S} d(o_i, o_j) - \epsilon$;

repeat

  foreach $\mathcal{D}_i$ *such that* $r'_i < r_i$ **do**

    if $(r'_i + d_{\min}) > r_i$ **then** $max\_inc \leftarrow r'_i - r_i$;

    else $max\_inc \leftarrow d_{\min}$;

    while $r'_i < max\_inc$ **do**

      find the smallest expansion e such that an event happens ;

      if $r'_i + e < max\_inc$ **then**

        $r'_i \leftarrow r'_i + e$;

        update $V^k(S')$ due to the event's consequences;

      else

        $r'_i \leftarrow max\_inc$;

        re-calculate edges/vertices corresponding to $\mathcal{D}_i$;

until *until all disks reach their targeted size*;

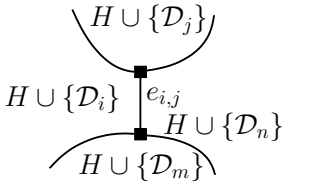
2.4.2 Insertion and deletion of sites in order-k max VDs

To insert a disk $\mathcal{D}_i(o_i, r_i)$ to an existing order-k max VD, we insert a disk $\mathcal{D}_{i'}$ centered at o_i with infinite radius, which does not change the diagram structure. Then, we shrink $\mathcal{D}_{i'}$ to point $\mathcal{D}_i$. It is easy to see that faces corresponding to $\mathcal{D}_{i'}$ only appear when $\mathcal{D}_{i'}$ contains less than k other disks. Updating the diagrams as $\mathcal{D}_{i'}$ shrinks can be done based on the fact that disk shrinking is an opposite process of disk expansion. From Lemmas 2.5 and 2.6, faces disappear as disks include other disks during their expansion. Therefore, when $\mathcal{D}_i$ excludes a disk from the last k disks interior to $\mathcal{D}_i$ during its shrinking, a face is born. Disk exclusions are asserted by sorting distances of other disk centers to o_i, which takes $O(N \log k)$ time. Shrinking $\mathcal{D}_i$ to p_i takes $O(kN \log N)$ time as shown in Algorithm 2.2. Thus, inserting a disk into an order-k max VD of N disks runs in $O(kN \log N)$.

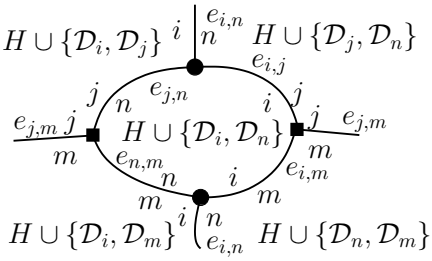
Deletion of a disk $\mathcal{D}_i$ from an existing order-k max VD is equivalent to expanding $\mathcal{D}_i$ such that it contains k other disks, which takes $O(kN \log N)$. We summarize the result in Theorem 2.3.

Theorem 2.3. *It takes $O(kN \log N)$ to insert a disk into or delete a disk from an order-k max VD of N disks.*

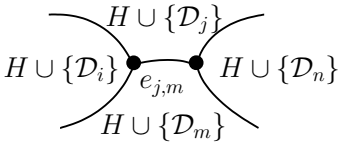
Note that, the result naturally extends to site insertion and deletion in order-k VD of points, which leads to the incremental construction of high order Voronoi diagrams of points in 2D. Dynamic data structure such as [15] and [55] can be used for updating the order-k VD as points are inserted, and querying the set of k points closest to the inserted point.



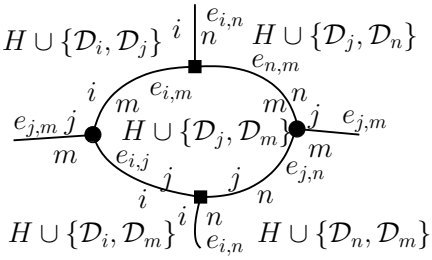
(a) $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i\}, S)$ is a neighbor of $\mathcal{V}^k(H \bigcup \{\mathcal{D}_n\}, S)$ in $V^{k-1}(S)$



(b) The corresponding order-k region of the structure in Figure a.



(c) $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j\}, S)$ is a neighbor of $\mathcal{V}^k(H \bigcup \{\mathcal{D}_m\}, S)$ in $V^{k-1}(S)$



(d) The corresponding order-k region of the structure in Figure c.

Figure 2.17: The evolution of faces $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_j\})$, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_m\}, S)$, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_n, \mathcal{D}_m\}, S)$, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_n\}, S)$, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_n\}, S)$, and $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S)$ ($|H| = k - 1$). Squares denote new vertices. Dots denote old vertices. Initially, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_n\}, S) \neq \emptyset$ and $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S) = \emptyset$. As $\mathcal{D}_n$ expands, $\mathcal{V}^k(H \bigcup \{\mathcal{D}_i, \mathcal{D}_n\}, S)$ shrinks, and eventually terminates, leading to the born of $\mathcal{V}^k(H \bigcup \{\mathcal{D}_j, \mathcal{D}_m\}, S)$.

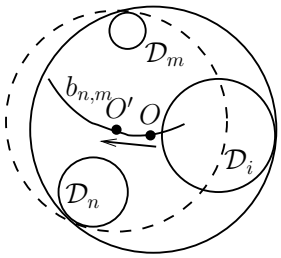


Figure 2.18: The expanding type-II disk $\mathcal{D}_i$ must first contact with a maximum circumference centered at a vertex.

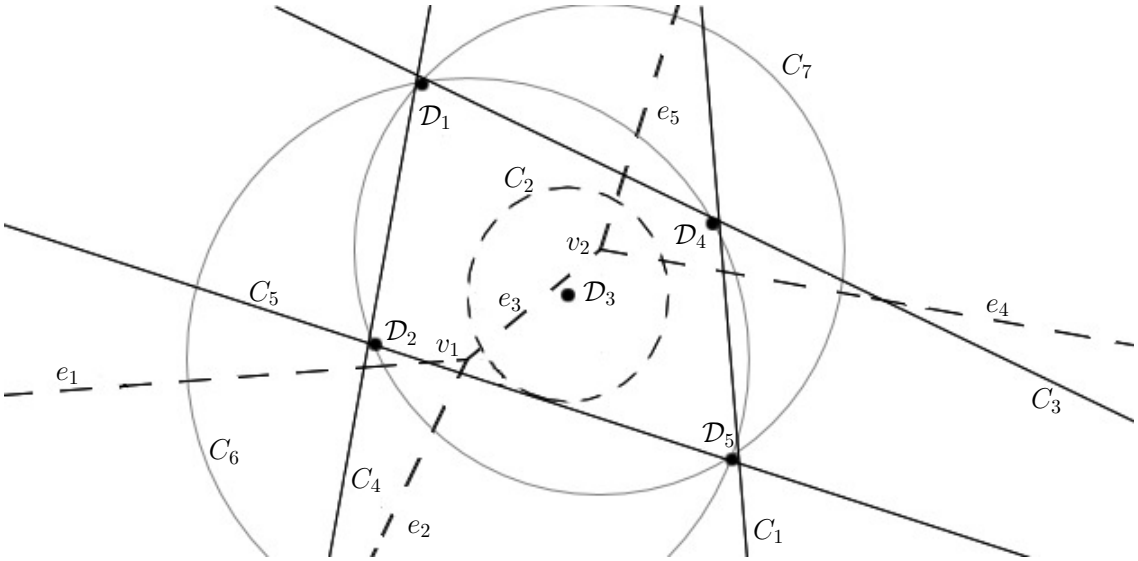


Figure 2.19: Illustration of a type-II disk's expansion. Dashed lines are the order-4 diagram of 5 points. Solid lines are maximum circumferences centered at infinite vertices.

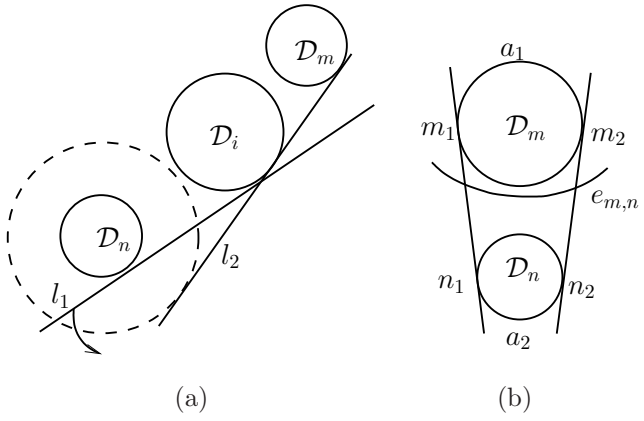
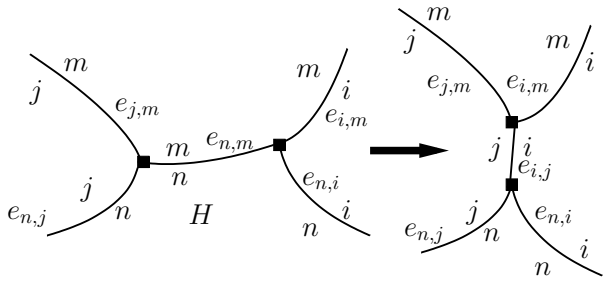
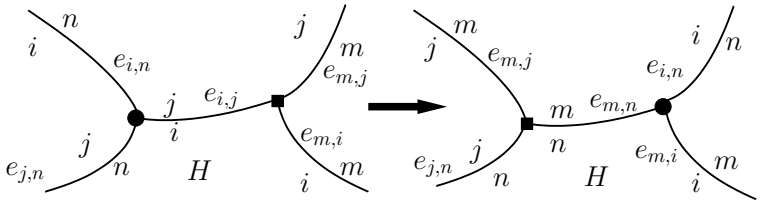


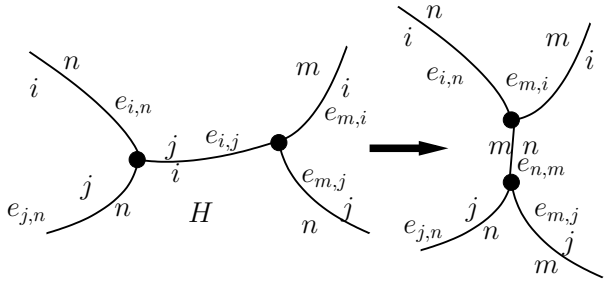
Figure 2.20: Illustration of Lemma 2.12's proof.



(a) Results of the meeting of 2 new vertices.



(b) Results of the meeting of a new vertex and an old vertex.



(c) Results of the meeting of 2 old vertices.

Figure 2.21: Changes of edges of $\mathcal{V}^k(H, S)$ as $\mathcal{D}_n$ expands.

Chapter 3

Sensor Location Uncertainties in Wireless Sensor Networks

3.1 Motivation

Wireless sensor networks (WSNs) have been employed in monitoring the health of civil structures, tracking targets, and studying the wild life habitat, among others. However, most existing work assumes knowledge of the *exact* location of the sensors. In practice, this assumption is rarely true. Even with on-board GPS receivers on all or selected nodes, or execution of distributed location algorithms [51], there generally exists uncertainty in sensor location. For instance, GPS receivers with WAAS (Wide Area Augmentation System) have an accuracy of three meters on average, considered to be a significant improvement over past generations which are accurate to within 15 meters on average. Even carefully positioned in the deployment phase, sensors may

be displaced due to environmental or human factors during the course of operation.

Uncertainty of sensor locations may also arise from privacy or security concerns. Consider the scenario where a sensor network (e.g, a group of GPS-enabled mobile devices) is formed to provide location services. Depending on the trustworthiness of its customers, the sensor network may only reveal coarse-grained vicinity information regarding the sensors' location and binary sensing measurements to its customer.

Modeling uncertainty regions deterministically has its root in the literature of robust optimization [51], where the uncertainty of the parameters in the objective function and constraints is modeled explicitly (as polyhedron, ellipsoid etc.). This is in contrast to stochastic optimization techniques, where the distribution of the uncertain parameters is assumed. Thus, robust optimization can be thought of as optimizing for the worst case. We feel the worst case analysis is suitable for sensor coverage in particular in security context though our approaches are notably different from robust optimization solutions due to their geometric nature.

One way to model the location uncertainty is by disks centered at the nominal location of the sensors. Under the boolean disk sensing model, a sensor is able to observe events located within a certain distance r to its location. In Chapters 2, we introduced the concept of order-k max Voronoi Diagram (VD) that tessellates the interested area into regions that are closer to k sensors in the worst case. In this chapter, we use it as a building block to design of efficient algorithms for various tasks in sensor networks, including locating and tracking targets, and network coverage. The tasks are considered under the disk uncertainty model.

Coverage and target localization and tracking are well-studied topics in wireless sensor networks. There has been a plural of work addressing the target localization and tracking in WSNs from both theoretical and system point of view [29, 39, 64, 65, 74, 82]. Coverage is a critical measure of the quality of sensing (QoS) in a wireless sensor networks. It indicates "how well do the sensors observe the physical space". There has been much work addressing the coverage problem in WSNs from both theoretical and system point of view [48, 44, 81]. In comparison to related work that deals with specific cases where either the sensor placement is regular (e.g., hexagonal) [34] or the uncertainty areas are identical [9], we consider generic settings of arbitrary placements, heterogeneous uncertainty areas, and polygonal area boundaries.

In this chapter, we provide a systematic analysis of two problems with consideration of sensor location uncertainties.

3.2 Network model

Consider n sensors $S = \{1, 2, \ldots, n\}$ in a bounded 2-D polygonal area region. Sensor i's location is arbitrarily distributed in a disk $\mathcal{D}_i(o_i, r_i)$ centered at its *nominal location* o_i with radius r_i. Uncertainty of sensor's locations is independent. We consider binary sensing and disk coverage, where each sensor i's sensing area is modeled as a disk of radius s_i centered at its *actual* location.

Given the *actual* location of a sensor i, a target at location p *can* be detected iff $d(i, p) < s_i$, where s_i is the sensing range. Consequently, given s_i, a location p is

guaranteed to be within the sensing range of sensor i iff $d_{\max}(p, \mathcal{D}_i) \leq s_i$. Similarly, p is *guaranteed* to be in the sensing range of k sensors if $\exists H \subset S$, $|H| = k$, such that $d_{\max}(p, \mathcal{D}_i) \leq s_i$, for all $i \in H$.

3.3 Robust coverage

3.3.1 Introduction

We utilize the order-k max VD in determining the minimum sensing radius needed to ensure worst-case k-coverage, call *k-exposure*. In comparison to related work that deals with specific cases where either the sensor placement is regular (e.g., hexagonal) [34] or the uncertainty areas are identical [9], we consider generic settings of arbitrary placements, heterogeneous uncertainty areas, and polygonal area boundaries.

3.3.2 Problem statements

When $r_i > 0$, to study coverage in the worst case, we define the maximum distance from a point p to a sensor $\mathcal{D}_i(o_i, r_i)$, as $d_{\max}(p, \mathcal{D}_i) = \max_{q \in \mathcal{D}_i} d(p, q)$ where $d(\cdot, \cdot)$ is the Euclidean distance. Clearly,

$$d_{\max}(p, \mathcal{D}_i) = d(p, o_i) + r_i.$$

Consequently, a location p is *guaranteed* to be covered by sensor i iff $d_{\max}(p, \mathcal{D}_i) \leq s_i$. Similarly, p is *guaranteed* to be k-covered iff $\exists H \subset S$, $|H| = k$, such that

$d_{\max}(p, \mathcal{D}_i) \leq s_i$, for all $i \in H$.

Guaranteed k-coverage is related to the notion of max VD and order-k max VD defined in Chapter 2 and repeated here for the sake of convenience.

Definition 3.1. *For i, $j \in S$, let $D(\mathcal{D}_i, \mathcal{D}_j) = \{p | d_{\max}(p, \mathcal{D}_i) < d_{\max}(p, \mathcal{D}_j)\}$. We define*

$$\mathcal{V}_k(S) = \bigcap_{i \in S, i \neq k} D(\mathcal{D}_k, \mathcal{D}_i)$$

the maximum-Voronoi region of $\mathcal{D}_k$ with respect to S. The maximum-Voronoi diagram (max VD) of S is defined as

$$V(S) = \bigcup_{i,j \in S, i \neq j} \mathcal{V}_i(S) \bigcap \mathcal{V}_j(S)$$

Definition 3.2. *Let $H \subset S, |H| = k$. We define*

$$\mathcal{V}^k(H, S) = \bigcap_{j \in H, i \notin H} D(\mathcal{D}_j, \mathcal{D}_i)$$

the order-k maximum-Voronoi region of a set of disks H with respect to S. The order-k maximum-Voronoi diagram of S is defined as

$$V^k(S) = \bigcup_{H, H' \subset S; H \neq H'; |H| = |H'| = k} \mathcal{V}^k(H, S) \bigcap \mathcal{V}^k(H', S)$$

Given $\mathcal{D}_i(o_i, r_i)$, $i \in S$, we are concerned with the max-min sensing radius of sensors so that a point p, or a polygonal area $\boldsymbol{B}$ is guaranteed to be covered. Formally,

Problem 3.1 (Guaranteed Point k-coverage)**.** *What is the minimum sensing range of the associated sensors in S so that a point $p \in \boldsymbol{B}$ is guaranteed to be k-covered.*

Problem 3.2 (Guaranteed Area k-coverage)**.** *What is the minimum sensing range of a set of sensors S so that any point p in the area $\boldsymbol{B}$ bounded by the boundary $\bar{\boldsymbol{B}}$ is k-covered by S.*

We will show in Section 3.3.3, that the sensing range for guaranteed point and area k-coverage can be determined from the order-k max VD. The max-min sensing radius in the area k-coverage is also called k-*exposure* due to its physical meaning as the radius of largest holes in the networks (in absence of sensors). In the remaining paper, we drop the term "guaranteed" when there is no ambiguity from the context.

Note that the max-Voronoi region of a disk $\mathcal{D}_i \in S$ does not always exist. Consider 2 disks $\mathcal{D}_i(o_i, r_i)$ and $\mathcal{D}_j(o_j, r_j)$. Suppose $r_i > r_j$, then the max-Voronoi region of $\mathcal{D}_i$ does not exist if $r_i - r_j > d(o_i, o_j)$. In this case, we refer to $\mathcal{D}_i$ as a *trivial disk*. In this work, we assume that no trivial disk exists. Furthermore, for ease of presence, we consider general configurations where no point in the plane has equal maximum distance to more than 3 sensors.

3.3.3 Solutions

Now we turn to the application of order-k max VD to address the two problems raised in Section 3.3.2, namely, guaranteed point k-coverage and area k-coverage. The objective is to determine the max-min sensing ranges in both problems.

3.3.3.1 Guaranteed point k-coverage

Given any point p in the interested area and $V(S)$, we first determine the set H of cardinality k, which are closer to p than any disk in $S - H$. Therefore, the minimum sensing range to ensure k-coverage to p is given by,

$$\gamma_k(p) = \max_{j \in H} \left\{ \max_{q \in \mathcal{D}_j} d(p, q) \right\} = \max_{j \in H} \{ d(o_j, p) + r_j \} \tag{3.1}$$

3.3.3.2 Guaranteed area k-coverage

Given a polygonal area $\boldsymbol{B}$ of interest, providing guaranteed k-coverage to $\boldsymbol{B}$ is equivalent to find a sensing range γ_k ensuring k-coverage to any point $p \in \boldsymbol{B}$. We show next, it suffices to consider only the vertices of $V^k(S)$, the intersections of $V^k(S)$ and $\bar{\boldsymbol{B}}$, and the vertices of $\bar{\boldsymbol{B}}$.

Given the set of sensors S, the area $\boldsymbol{B}$ is tessellated into faces $\mathcal{F}_k(H)$ for $|H| = k$ and $\mathcal{V}^k(H, S) \neq \emptyset$. The boundary of $\mathcal{F}(H)$ consists of hyperbolic arcs due to order-k max VD edges (or a subsegment), and line segments in $\bar{\boldsymbol{B}}$.

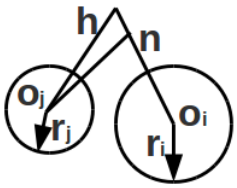
Lemma 3.1. *Let T and U be the set of vertices of the boundary $\bar{\boldsymbol{B}}$, and intersections of $\bar{\boldsymbol{B}}$ and $V^k(S)$, respectively. The area $\bar{\boldsymbol{B}} \bigcup \boldsymbol{B}$ is k-covered iff all points in T, U, and Voronoi vertices of $V^k(S)$ are k-covered for sensing range γ_k.*

Proof. The necessary condition is obvious. To prove the sufficient condition, we prove 2 arguments: 1. Every point in $\mathcal{F}(H)$ is k-covered if all points in the the boundary of $\mathcal{F}(H)$, $\bar{\mathcal{F}}$, is k-covered; and 2. All points on the boundary $\bar{\mathcal{F}}$ of $\mathcal{F}(H)$ are k-covered if all vertices of $V^k(S)$ in $\bar{\mathcal{F}}$, and points in $T \bigcup U$ which are in $\bar{\mathcal{F}}$ are k-covered, where

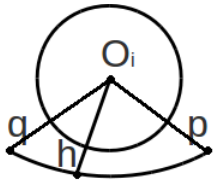
$H \subset S$ and $\mathcal{F}(H) \neq \emptyset$. To prove argument 1, let n be a point $\in \mathcal{F}(H)$, and let $\mathcal{D}_i \in H$ such that $d_{\max}(p, \mathcal{D}_i) \geq d_{\max}(k, \mathcal{D}_j)$, $\forall\ \mathcal{D}_j \in H$. Let $h = o_i n \bigcap \bar{\mathcal{F}}$. We have $d_{\max}(h, \mathcal{D}_i) = d_{\max}(n, \mathcal{D}_i) + d(h, n) \geq d_{\max}(n, \mathcal{D}_j) + d(h, n) \geq d_{\max}(h, \mathcal{D}_j)\ \forall\ \mathcal{D}_j \in H$. The last inequality is due to the properties of triangle inequalities (as illustrated in Figure 3.1a). Therefore, if h is k-covered then so is n. Clearly, $\exists h \in \bar{\mathcal{F}}$ for $\forall p \in \mathcal{F}(H)$, $\mathcal{F}(H)$ is k-covered when $\bar{\mathcal{F}}$ is k-covered. To prove the latter argument, we first make the following observation. Assume $h \in$ some edge of $pq \subset \bar{\mathcal{F}}$. It can be shown that $\exists \mathcal{D}_i \in H$ such that $d_{\max}(h, \mathcal{D}_k) \leq d_{\max}(h, \mathcal{D}_i) < \max\{d_{\max}(p, \mathcal{D}_i), d_{\max}(q, \mathcal{D}_i)\}$, $\forall \mathcal{D}_k \in H$, based on the properties of max-VD edges as discussed in Chapter 2, and on the properties hyperbolas and triangles as illustrated in Figures 3.1b - 3.1c. Now, assume that $pq \subset \mathcal{V}^k(H, S)$, and that $d_{\max}(p, \mathcal{D}_i) \geq d_{\max}(p, \mathcal{D}_k)$ and $d_{\max}(q, \mathcal{D}_j) \geq d_{\max}(q, \mathcal{D}_k)$, $\forall \mathcal{D}_k \in H$. Obviously, $\gamma_k = \max\{d_{max}(p, \mathcal{D}_i) d_{\max}(q, \mathcal{D}_j), \}$. W.l.o.g, assume that $d_{\max}(q, \mathcal{D}_j) \geq d_{\max}(p, \mathcal{D}_i)$. Based on the observation, $d_{\max}(q, \mathcal{D}_k) \leq d_{\max}(q, \mathcal{D}_j) \leq \max\{d_{\max}(p, \mathcal{D}_j), d_{\max}(q, \mathcal{D}_j)\} < \max\{d_{\max}(p, \mathcal{D}_i), d_{\max}(q, \mathcal{D}_j)\} = \gamma_k$, $\forall \mathcal{D}_k \in H$. Thus, if p, q are k-covered, then h is k-covered. Since the two arguments is true for all regions of $V^k(S)$, the lemma is proved. □

From Lemma 3.1, we can easily devise a procedure to compute k-exposure, i.e., the max-min sensing range for area k-coverage as outlined in Algorithm 3.1.

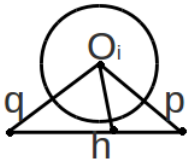
By Chapter 2, the cost of line 1 is $O(k^2 \cdot T(n))$, where $T(n)$ is the time to compute the order-1 max VD. Let m be the number of vertices on $\bar{\boldsymbol{B}}$. The cost of finding $\gamma_k(p)$, when p is a vertex on $\bar{\boldsymbol{B}}$ is $NNQ(n, k)$, the cost of nearest neighbor query on $V_k(S)$. The cost is constant for Voronoi vertices in $V_k(S)$, and intersection of $V(S)$ and $\bar{\boldsymbol{B}}$, and there are $O(kn)$ of them. Thus, we have the following bound on



(a) Inequality in triangle: $d(o_j, h) < d(o_j, n) + d(n, h)$.



(b) $h \in pq$, a hyperbolic arc with focus o_i: $d(h, o_i) \leq \max\{d(o_i, q), d(o_i, p)\}$.



(c) $h \in pq$ of the triangle $o_i pq$: $d(o_i, h) \leq \max\{d(o_i, q), d(o_i, p)\}$.

Figure 3.1: Illustration for the proof of Lemma 3.1.

Algorithm 3.1: Minimum sensing range for k-coverage under uncertainty of sensors' locations

---

input : Set of sensors $S = \{\mathcal{D}_1(o_1, r_1), \ldots, \mathcal{D}_n(o_n, r_n)\}$, the area boundary $\boldsymbol{B}$, and $k \geq 1$.

output: The minimum sensing range $\mathcal{R}$ so as to guarantee the k-coverage in $\bar{\boldsymbol{B}}$

1 Compute $V^k(S)$, the order-k max VD of S;

2 $\mathbb{V} \leftarrow$ vertices of $V^k(S)$ in $\boldsymbol{B}$;

3 $\mathbb{V} \leftarrow \mathbb{V} \bigcup \left(V^k(S) \bigcap \bar{\boldsymbol{B}}\right) \bigcup$ vertices of $\bar{\boldsymbol{B}}$;

4 $\gamma_k \leftarrow \max_{p \in \mathbb{V}} \gamma_k(p)$, where $\gamma_k(p)$ is computed from (3.1).

---

computing k-exposure.

Theorem 3.1. *Given S and a m-vertex polygonal area B. The total cost of calculating k-exposure of B, γ_k, is $O(k^2 \cdot T(n) + mNNQ(n,k) + kn)$, where $T(n)$ is the cost of constructing the order-1 max VD of n disks, and $NNQ(n,k)$ is the cost of k nearest neighbors query on $V^k(S)$.*

3.3.4 Results

We have implemented our algorithm in CGAL [1], a computation geometry library. In this section, we demonstrate the correctness of the proposed algorithms and evaluate k-exposure for randomly deployed sensors.

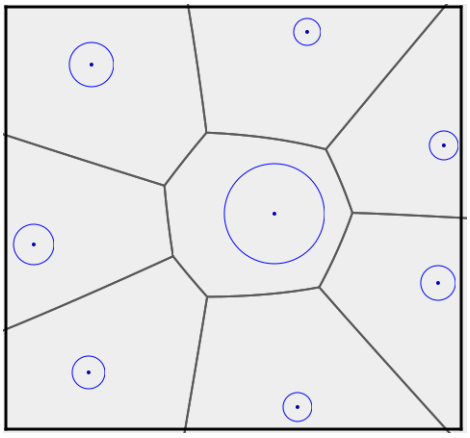
We compare the k-exposure determined by Algorithm 3.1 with that by a naive approach that utilizes the regular point VD. In the naive approach, for $S' = \{o_1, o_2, \ldots, o_n\}$, we first constructs order-1 VD and then apply Lee's algorithm [43] to construct order-k VD for S'. Next, for each Voronoi vertex v in $\boldsymbol{B}$, we compute the distance to the corresponding three sites, say, $\{o_i, o_j, o_l\}$ and set $\gamma'_k(v) = d(v, o_i) + \max\{r_i, r_j, r_l\}$. For each non-vertex v on $\bar{\boldsymbol{B}}$, we find its k nearest neighbors on order-k VD and compute the $\gamma'_k(v)$ similarly. Finally, the k-exposure is found by taking the maximum among all $\gamma'_k(v)$'s. We claim without proof that this procedure determines a sensing range that ensures k-coverage under uncertainty. However, it is necessarily equal or bigger than the k-exposure determined by Algorithm 3.1.

Toy examples To understand the difference between the k-exposure determined by the proposed and the naive approach, we first consider toy scenarios in Figure 3.2a and Figure 3.2c. In the figures, the nominal locations and uncertainty region of sensors are represented by dot and disks, respectively. Note that the variability among uncertainty regions of sensors are higher in Figure 3.2c. Figure 3.2b and 3.2d compare the 1-exposure found by both methods. We can see that while the exposure determined by our algorithm is strictly not greater, the difference between the two is more salient when the variability in the uncertainty regions is higher.

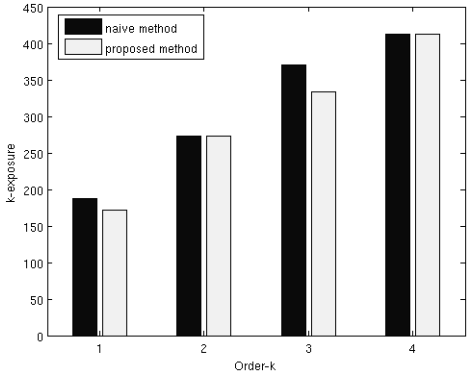
A realistic scenario Next, we consider a more realistic scenario, in which 23 sensors are randomly deployed in a 500×320 field. The uncertainty radii are generated uniformly between 20 and 60.

Figure 3.3(a)–(f) show the order 1 through 3 max VD and their respective exposures. In Figure 3.3(d)–(f), we include disks that are centered at the nominal location of each sensor. Clearly, the desired degree of coverage is provided. One interesting quantity is the ratio of k and $(k+1)$-exposure. The bigger the ratio the higher the redundancy the placement for k-coverage. Some points in the area are in fact $(k+1)$-or-higher covered when the sensing range is set to be the k-exposure. Our experiments show that as the sensing range is set to be 157.23. Roughly, 90% of the area are 2-or-higher covered.

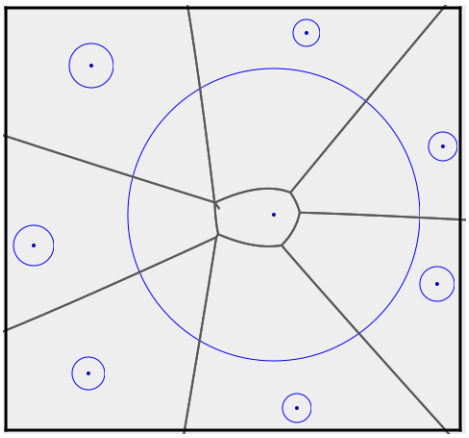
Figure 3.4 compares the exposure determined by the two algorithms in 30 random deployments of 25 sensors in a 500×320 field. Uncertainty disks are uniformly selected in 3 different ranges, namely, $[0, 20]$, $[0, 40]$, and $[0, 60]$. As seen in the



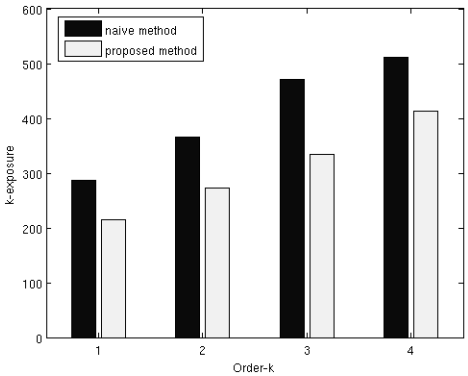
(a) Low-variance uncertainties.



(b) Performance of the two approaches in low-variance uncertainties.

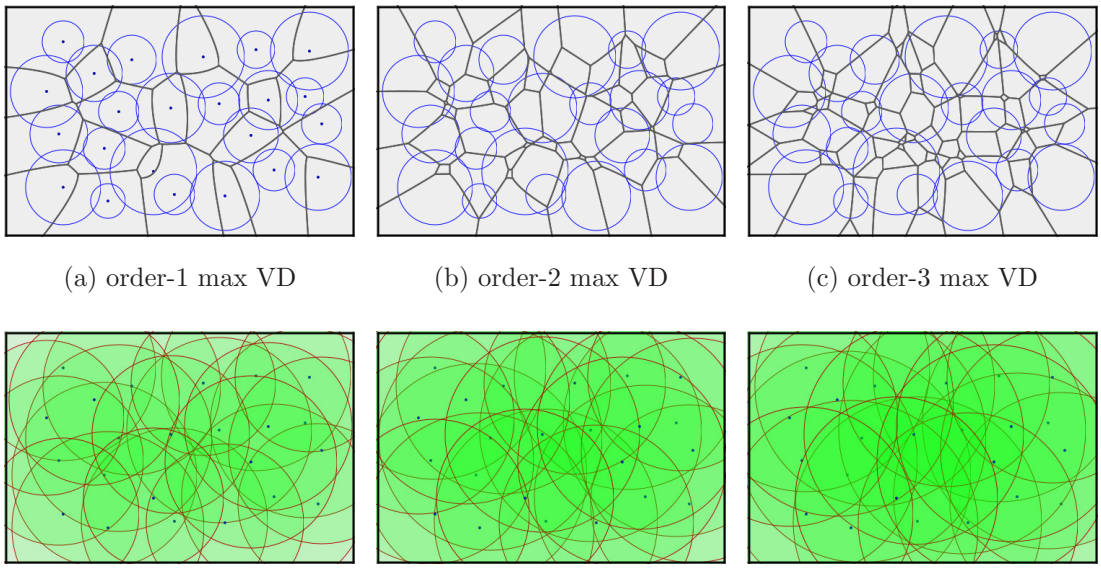


(c) High-variance uncertainties.



(d) Performance of the two approaches in high-variance uncertainties.

Figure 3.2: Compare of the proposed method with the naive one in various sensors' uncertainty in different orders. In Figures 3.2a and 3.2c, disks present uncertainties of sensors' locations, solid arcs present the order-1 max VD. In Figures 3.2b and 3.2d, the x-axis presents the order of the max VD, the y-axis shows the k-exposures of orders 1-4.



(a) order-1 max VD (b) order-2 max VD (c) order-3 max VD

(d) 1-exposure = 157.23 (e) 2-exposure= 183.57 (f) 3-exposure = 200.19

Figure 3.3: 1-, 2-, and 3-coverage of 23 sensors randomly deployed in the area of 500×320. Dots are sensors' nominal locations. In Figure 3.3a - 3.3c: disks are uncertainty areas, dark arcs are high order max VDs. Solid disks in Figure 3.3d - 3.3f are exposures.

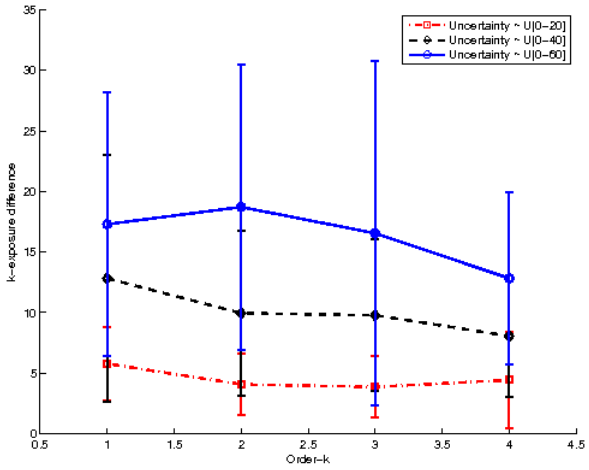


Figure 3.4: Performance of the proposed method and the naive approach on 25 sensors randomly deployed in a 500×320 field. The means and variances of exposure deviations of our proposed method and the naive one under different orders $(1-4)$ and uncertainty variances are shown.

figure, the higher the uncertainty variance, the smaller the proposed method's k-exposure compared to the naive one's. Finally, we vary the number of sensors and measure its impact on the k-exposure. The results are shown in Figure 3.5. As expected, as the number of sensors increases, the k-exposure decreased for all orders.

3.4 Related work

The objective of k-coverage ($k \geq 1$) is to place sensors at locations such that the union of the sensing disks cover the entire area of interest. For 1-coverage, placing homogeneous sensors in a hexagon grid pattern is known to be optimal [34]. However,

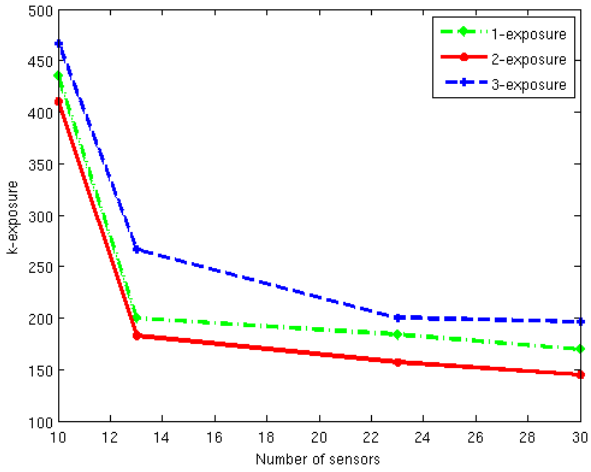


Figure 3.5: k-exposure VS. number of sensors. Sensors are randomly deployed in a 500×320 field.

such a placement is very sensitive to the location of sensors. A small movement of one sensor from its designated grid position leaves some region uncovered. On the theoretical front, it has been shown in [81] that to provide k-coverage with high probability, the sensor density needs to scale on the order of $2k \log\log A$, where A is the size of the area if sensors are deployed following a Poisson point process. Again, it is assumed that sensor locations are exact.

Much work has been done to detect coverage and communication holes in wireless sensor networks, which can be broadly categorized into two groups, namely 1) those utilizing location information and 2) those utilizing connectivity information. Fang *et al.* [19] proposed a simple algorithm that greedily sweeps along hole boundaries and discovers boundary cycles. Location information is assumed to be known and unit disk connectivity is required. Ghrist and Muhammad [25] proposed a centralized

method to detect coverage holes by means of homology utilizing network communication graphs. Assuming that the connectivity is determined by the unit-disk graph model, Funke and Klein [22] proved that using linear-time algorithm, one can identify nodes on the boundaries of holes of the network. Wang *et al.* [75] developed a practical distributed algorithm for boundary detection in sensor networks, using only the communication graph. The basic idea is to identify irregularity in the communication graph, more specifically "cut" in shortest path tree, whose common ancestor is far away. Holes in communication graphs can have different definitions based on the application context. To form a hole, the diameter of the holes is larger than communication range.

When there exists uncertainty in sensor location, two classes of problems have been considered in literature: 1) positioning sensors given bounds on the placement error and 2) determining the amount of room for error given a chosen set of intended sensor positions under some restrictive conditions. Bar-Noy *et al. [9] established the connection between inexact sensor location modeled as a disk of radius w and shrinking the sensor range from R to $R - w$ under deterministic and probabilistic sensing models. Under such mapping, existing techniques for sensor placement and the inverse problem of sensing radius determination can be applied. The authors described an $O(n \log n)$ complexity algorithm for radium assignment based on Voronoi diagrams. The proposed method is a special case of our approach when the uncertainty area of sensors is identical.*

In [34], Johnson et al. considered regular grid deployment and investigate the benefit of shrinking the grid size to remedy the uncertainty in sensor locations. They

show that shrinking the grid size to obtain a denser hexagonal lattice allows all sensors to move about their intended positions independently while nonetheless guaranteeing full coverage. Furthermore, sufficiently increasing the lattice density will naturally yield k-coverage for $k > 1$. Wiggle room was used to characterize the allowance of uncertainty in sensor position departing from the ideal hexagonal placement to ensure coverage. Our work differs from [34] in that the uncertainty area (or wiggle room) is given while the objective is to determine whether the area is fully covered. Furthermore, our methods target for general configurations and are not limited to hexagonal placement.

3.5 Target localization and tracking

3.5.1 Introduction

We design of efficient algorithms for locating and tracking targets using binary proximity sensors with location uncertainty. In particular, for target localization without sensing errors, an average running time of $O(k^2 \log k)$ is incurred, where k is the number of sensors detecting the target. Evaluating the likelihood of a candidate location in presence of sensing errors has an average time complexity of $O(\log N + k)$ given a total of N sensors. Both are considerably more efficient than naive approaches. The likelihood estimates play a central part in evaluating the measurement model to handle both sources of uncertainties, and when combined with filter-based approaches, give rises to efficient algorithms for target tracking.

3.5.2 Problem statements

Let $C(p, s_i)$ be the disk centered at the target location p with radius s_i, U_i be the overlapping area of $C(p, s_i)$ and $\mathcal{D}_i$. With the actual sensor locations uniformly distributed in the uncertainty disks, the binary measurement $z_i(t)$ of sensor i at time t with respect to target p follows a Bernoulli distribution given by:

$$\begin{aligned}\mathcal{P}(z_i(t)|p) = &\left(\frac{\mathcal{A}(U_i)}{\mathcal{A}(\mathcal{D}_i)}(1-\epsilon_i) + \frac{\mathcal{A}(\mathcal{D}_i)-\mathcal{A}(U_i)}{\mathcal{A}(\mathcal{D}_i)}\epsilon_i\right) z_i(t) \\ &+ \left(\frac{\mathcal{A}(U_i)}{\mathcal{A}(\mathcal{D}_i)}\epsilon_i + \frac{\mathcal{A}(\mathcal{D}_i)-\mathcal{A}(U_i)}{\mathcal{A}(\mathcal{D}_i)}(1-\epsilon_i)\right)(1-z_i(t)) \end{aligned} \tag{3.2}$$

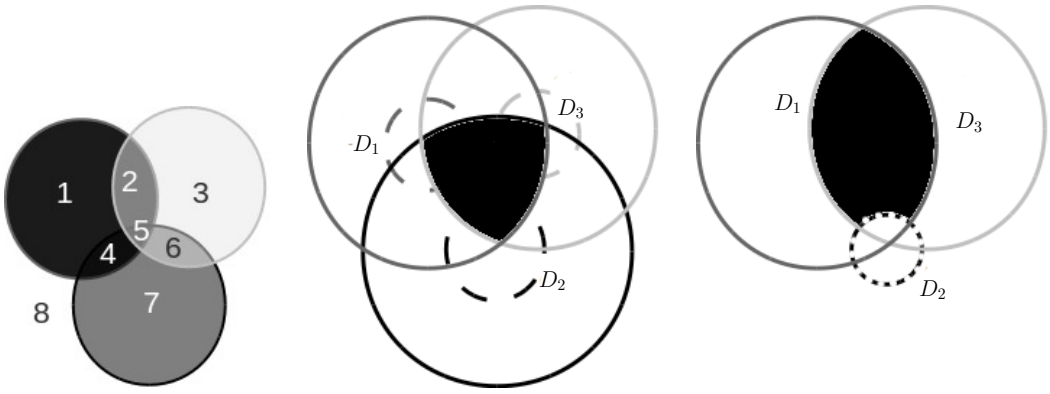
where $\mathcal{A}(\cdot)$ denotes the area. When $\mathcal{A}(U_i) = 0$ (i.e., sensor i cannot detect targets at location p), the above equation can be simplified as,

$$\mathcal{P}(z_i(t)|p) = \epsilon_i z_i(t) + (1-\epsilon_i)\,(1-z_i(t)) \tag{3.3}$$

In this chapter, we consider the problems of target localization and tracking in a field of binary proximity sensors with measurement and location uncertainty. Formally,

Problem 3.3. (Target localization with no measurement errors) *Given a binary vector of length N, $Z(t)$, of the readings from N sensors, assuming no measurement errors, what is the region that a target MAY be located?*

It is well-known that the binary proximate sensors only provide a coarse-grained spatial resolution. In fact, it has been proven in [64] that if a network of binary proximity sensors has an average sensor density ρ and each sensor has sensing radius



(a) Tessellation due to 3 sensors (no location uncertainty).

(b) The localized area as sensors $\mathcal{D}_1, \mathcal{D}_2$, and $\mathcal{D}_3$ detect a target.

(c) The localized area as only sensors $\mathcal{D}_1$ and $\mathcal{D}_3$ detect a target.

Figure 3.6: Ambiguity of target locations with ((b)(c)) and without (a) sensor location uncertainty. The solid, dashed and dotted circles enclose areas of probable detection, sensor uncertainty region and guaranteed detection, respectively

s, then the worst-case L_∞ error in localizing the target is at least $\Omega(1/\rho s)$. With the uncertainty of sensor locations, the ambiguity is even more significant. To see why, let us consider an illustrative example with three sensors in Figure 3.6. Figure 3.6(a) shows the likely target locations from different sensor readings when the locations of sensors are exact. For instance, region 5 corresponds to the reading $z_1 \wedge z_2 \wedge z_3 = 1$ while region 2 corresponds to $z_1 \wedge z_3 \wedge \neg z_2 = 1$. Figures 3.6(b) and (c) show the regions for $z_1 \wedge z_2 \wedge z_3 = 1$ and $z_1 \wedge z_3 \wedge \neg z_2 = 1$, respectively. In Figures 3.6(a)(b), the dashed disks represent the uncertainty regions, the solid disk has radius $r_i + s_i$ for sensor i (the probable region of targets), and the dotted disk has radius $\max\{s_i - r_i, 0\}$ (the guaranteed region of target). For instance, any point in the shaded region in Figure 3.6(b) may be detected by all sensors. As sensors 1 and 3 detect targets but

sensor 2 does not, targets locate in the shaded region in Figure 3.6(c).

A few observations are in order. First, when there is no uncertainty in sensor locations, the exclusion-inclusion of sensing disks around sensors form a tessellation of the entire area. This can be used as the basis for localization by simply labeling each region a the binary observation vector. However, such a property no longer holds with location uncertainty in sensors, namely, the areas corresponding to different readings may overlap with one another. A naive approach would incur $O(n \log n)$ *complexity to compute each possible region by a sweep-line algorithm [62]. Second, expanding the sensing range by the uncertainty radius would not yield proper localization results. The reason is not only we need to consider inclusion of these expanded disks due to sensors with positive readings, but also exclusion of reduced disks (with radius equal to sensing radius minus the uncertainty radius) due to sensors with null readings.*

Problem 3.4. (Likelihood of observations given a target location) *Let* Z *be the binary measurement vector (possibly with errors). What is the likelihood of having reading* Z *for a target at location* p*?*

Equation (3.2) *can be used to evaluate the above probability. However, to compute the* U_i*'s, a naive approach would inspect every single sensor and test whether* U_i *is empty or not resulting in a complexity of* $O(N)$*. We will show that only a small set of sensors need to be tested with the help of geometric structures defined in Section 2. More importantly, we will show an interesting result that the average running time of the localization algorithm only depends on the number of sensors detecting targets.*

Problem 3.5. (Single target tracking with measurement errors and

LOCATION UNCERTAINTY) *Given sensor observations* $Z(t_0), Z(t_1), \ldots, Z(t_n)$ *at times* $t_0, t_1, \ldots, t_n$*, what is the trajectory of the target?*

Clearly, due to limited spatial resolution for target locations, target trajectories cannot be uniquely determined as well. Additional constraints need to be imposed. Shrivastava et al. in [64] applies the Occam's razor principle to trim the target trajectory. We choose to apply a particle filter-based approach.

3.5.3 Solutions

In this section, we first discuss the K-nearest neighbor queries, which play as the foundation for the Problem 3.4. Then, we give the solution to the problems formulated in the previous section.

3.5.3.1 K-nearest-neighbor (kNN) query

kNN queries concern with finding the k nearest neighbors among N disks of a location in the plane under $d_{\max}$ *distance metric. Order-k max VDs help to improve the running time of kNN queries. The main results of this section are summarized as follows:*

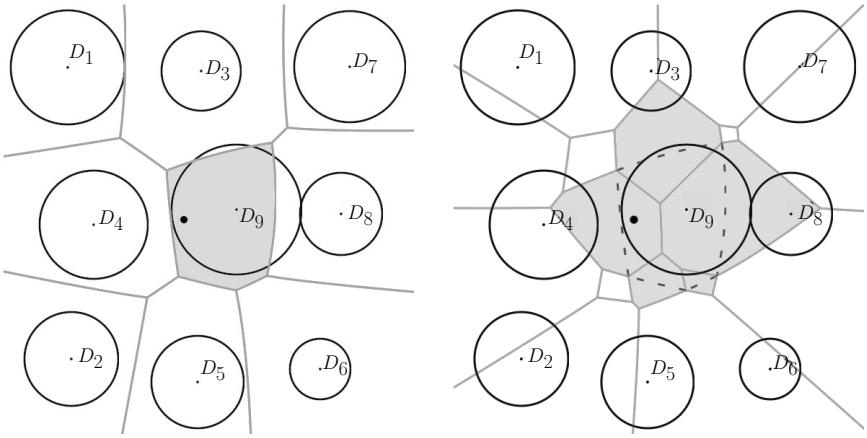
Theorem 3.2. *Given an order-k max VD of a set S of N disks, a single kNN query takes* $O(\log N)$ *and* $O(N)$ *running time complexity on average, and in the worst cases, respectively.*

Theorem 3.3. *Given the max VDs of orders* $1, 2, \ldots, N-1$ *of* N *disks,* K *sequential queries, i.e., finding the* $1, 2, \ldots, K$ *nearest neighbors takes* $O(N \log N)$ *and* $O(\log N + K)$ *times in the worst case, and on average, respectively.*

kNN query *We replace every finite edge (curve) in the order-k max VD* $V^k(S)$ *by a straight line segment connecting its vertices, and replace half-edges (with a single vertex) or open edges (no end points) a ray or an infinite straight line that do not intersect other rays, lines or line segments. This procedure results in a planar subdivision* $\mathcal{D}$*, in which there is an one-to-one mapping from a particular region* $\mathcal{V}$ *of* $V^k(S)$ *to a region* $\mathcal{R}$ *of* $\mathcal{D}$*. From the construction, a point in a* $\mathcal{R}$ *must locate in* $\mathcal{V}$ *or* $\mathcal{V}$*'s neighbor regions. Determining the region* $\mathcal{D}$ *containing the query point takes* $O(\log n)$ *[40]. Since two neighboring regions differ by one disk, it takes* $O(1)$ *to determine whether the query point is in* $\mathcal{V}$ *or one of its neighboring regions. Since the average number of neighbors of* $\mathcal{V}$ *in* $V^k(S)$ *is constant ([7]), finding the region of* $V^k(S)$ *that contains a given point takes* $O(\log N)$ *time on average with the worst-case time* $O(N)$*. We have Theorem 3.2.*

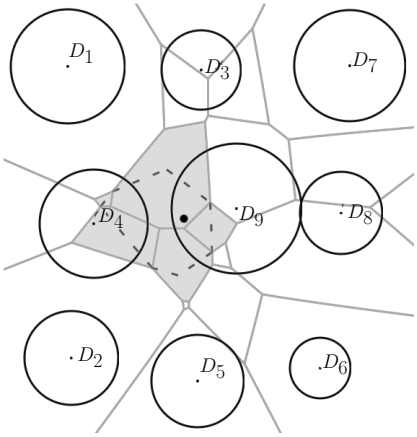
kNN queries *A straightforward approach for finding the* $1, 2, \ldots, K$*,* $(K > 0)$ *nearest neighbors takes* $O(K \log N)$ *by applying the single kNN query* K *times (with* $k = 1, 2, \ldots, K$*). We show next that by leveraging the max VDs up to the* K*th order, the complexity can be greatly reduced.*

Lemma 3.2. *Consider a point* p *in a particular region* $\mathcal{V}^k(H, S)$ *of* $V^k(S)$*, that is,* H *is the set of* k *nearest neighbors of* p*. Let* $H' = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_l\}$ *be the set of disks*



(a) Order-1 max VD

(b) Order-2 max VD



(c) Order-3 max VD

Figure 3.7: Sequential kNN query of a location (dot) in the first 3 high-order max VDs of 9 disks (circles). In order-1 max VD (a), the location is queried in the whole plane. Once the region of order-k is identified (dashed), only the regions of the order-$(k+1)$ VD that tessellate the order-k region (shaded) are searched (b)(c).

associated with the new vertices of $\mathcal{V}(H, S)$. *The* $(k+1)^{th}$ *nearest disk of* p *must be an element of* $H' \backslash H$.

Given the k nearest neighbors of a point p, Lemma 3.2 limits the number of candidates for the $(k+1)^{st}$ nearest neighbor. Since $\mathcal{V}^k(H,S)$ has on average 6 neighbors, the number of new vertices of $\mathcal{V}(H,S)$ is also constant, and so is the total number of disks associated with the new vertices of $\mathcal{V}^k(H,S)$. Therefore, the number of probable $(k+1)^{st}$ nearest disks is constant. Figure 3.7 illustrates the incremental algorithm for the sequential kNN queries.

Algorithm 3.2: Sequential kNN queries

Input : A set of disks $S = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_1(o_2, r_2), \ldots, \mathcal{D}_N(o_N, r_N)\}$, order-$k$ max VDs $V^1(S), V^2(S), \ldots, V^{N-1}(S)$, a query point p.

Output: Sorted K nearest disks of p.

1 $\mathcal{V}(H_1, S) \leftarrow$ *the order-1 max Voronoi region that contains* p ;

2 $L \leftarrow H_1$ `/* List of sorted` K `nearest disks of` p `*/`;

3 **for** $k \leftarrow 2$ **to** K **do**

4 $\quad \mathbb{V} \leftarrow$ new vertices of $\mathcal{V}(H_{k-1}, S)$;

5 $\quad \mathbb{H} \leftarrow$ disks associated with $\mathbb{V}$, which are not in H_{k-1} ;

6 $\quad \mathcal{R} \leftarrow \left\{\mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S)\right\}, \forall \mathcal{D}_i \in \mathbb{H}$;

7 $\quad \mathcal{V}(H_k, S) \leftarrow \mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S) \in \mathcal{R} : p \in \mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S)$;

8 $\quad L \leftarrow L \bigcup \mathcal{D}_i$;

9 return L;

Algorithm 3.2 gives details of sequential queries. The algorithm first identifies the nearest neighbor of p (line 1) making use of the single 1NN query primitive in the previous section with $O(\log N)$ complexity on average. To find the k^{th} nearest disk ($k > 1$), the algorithm only searches among the neighbors of the region identified

in the $(k-1)^{st}$ step (lines 6 and 7). Since there are only a constant number of neighboring regions, line 7 takes $O(1)$ running time. Thus, we have the average running time as shown in Theorem 3.3. Additionally, let n_i be the number of disks in $\mathbb{H}$ (line 5) at the i^{th} iteration, the total number of comparisons up to the j^{th} iteration is $\sum_{i=1\ldots j} n_i$. If the sum exceeds $O(N \log N)$, we can switch to the naive method, i.e., sorting the distances from p to disks $j^{th}, (j+1)^{st}, \ldots, N$, which gives a worst case complexity of $O(N \log N)$. This proves the worst case complexity in Theorem 3.3. Note this procedure can be straightforwardly applied to answer $l+1, l+2, l+K$-NN queries for any $l \geq 0$.

To this end, we have developed the construction algorithm for order-k max VDs and algorithms for single and sequential kNN query. These primitives will be used as building blocks to address the three problems raised in Section 3.5.2.

3.5.3.2 Target localization with uncertainty

In this section, we consider the first problem of identifying the region that a target may be located given error-free binary readings from N sensors. Let H be the set of sensors detecting the target. In the geometric terms, given sensing range s_i's, we are seeking $\mathcal{R}$, the locus of points p satisfying

$$d_{\min}(p, \mathcal{D}_i) \leq s_i \text{ and } d_{\max}(p, \mathcal{D}_j) > s_j, \forall \mathcal{D}_i \in H, \mathcal{D}_j \notin H. \tag{3.4}$$

Without loss of generality, we set $s_i \equiv s$. Define $\mathcal{B} = \bigcup_{\mathcal{D}_j \in S \setminus H} \mathcal{D}_j(o_j, \max(s - r_j, 0))$

and $\mathcal{A} = \bigcap_{\mathcal{D}_i \in H} \mathcal{D}_i(o_i, r_i + s)$. *We first observe that,*

$$\mathcal{R} = \mathcal{A} \cap \neg\mathcal{B} \tag{3.5}$$

Essentially, we try to find a region that is inside the set of disks $C_i(o_i, s + r_i)$ *for* $i \in H$ *and outside the disks* $C_j(o_j, \max(s - r_j, 0))$ *for* $j \in S \backslash H$. *Further note that by the definition of* $d_{\min}$ *and* $d_{\max}$, *a necessary condition for* (3.4) *is*

$$d(p, o_i) - r_i \leq d(p, o_j) + r_j, \forall \mathcal{D}_i \in H, \mathcal{D}_j \notin H.$$

Or, equivalently,

$$d(p, o_i) - r_i + r_{\max} \leq d(p, o_j) + r_j + r_{\max}, \forall \mathcal{D}_i \in H, \mathcal{D}_j \notin H,$$

where $r_{\max} = \max_i r_i$. *Let* $r'_i = r_{\max} - r_i$ *if* $\mathcal{D}_i \in H$, *and vice versa. With the uncertainty radii properly adjusted and together with* (3.5), *we have*

$$\mathcal{R} \subseteq \mathcal{A} \cap \mathcal{V}^k(H, S),$$

where $k = |H|$.

Let H_i, $i = 1, \ldots, l$ *be the neighbors of* H *with common edges in* $\mathcal{V}^k(H, S)$ *and* $S' = \bigcup_i^l H_i \cup H$. *It is easy to show that* $\mathcal{V}^k(H, S) = \mathcal{V}^k(H, S')$. *Therefore, it suffices to compute* $\mathcal{B}' = \bigcup_{\mathcal{D}_j \in S' \backslash H} \mathcal{D}_j(o_j, \max(s - r_j, 0))$ *and* $\mathcal{R} = \mathcal{A} \cap \mathcal{B}'$.

Algorithm 3.3 gives details of target localization. It consists of two main steps, i.e., the construction of an order-k max VD (line 6), and disk inclusion-exclusion (line 8). We observe from the order-k max VD construction from Chapter 2 that the running time of expansion/shrinking of a disk in fact involves in the sorting to edges

of the Voronoi regions corresponding to the disk. Since the expected number of edges in a Voronoi region is constant, and the expected number of regions corresponding to a disk in an order-k max VD is k, expansion/shrinking N disks in an order-k max VD takes $O(Nk \log k)$ expected running time. The order-k max VD in this case can be constructed by shrinking disks corresponding to sensors detecting targets. Thus, the average running time of line 8 is $O(k^2 \log k)$. To see the average running time of the second step, note that $|H' \backslash H|$ in fact is the number of neighbor regions of $\mathcal{V}(H, S)$, which is constant on average. Therefore, the inclusion and exclusion (line 8) takes $O(k \log k)$. We conclude:

Theorem 3.4. *Given the readings of N sensor in which k sensors detect the target, Algorithm 3.3 on average takes $O(k^2 \log k)$.*

3.5.3.3 Likelihood of sensor readings under noisy measurements

Given noisy binary observation Z and a candidate target location p, we are interested in evaluating its likelihood under the measurement model in (3.2)*. This requires calculating the overlapping area U_i of $C(p, s)$ and $\mathcal{D}_i$ for $i = 1, 2, \ldots, N$, some of which can be empty (Figure 3.8). A naive approach would compute the distance from p to every $\mathcal{D}_i$ costing a total computation time of $O(N)$. With the help of the kNN queries developed in Section 3.5.3.1, we will show that a more efficient algorithm can be devised.*

The basic idea is to identify the set of sensors H that are possible to detect p. Formally, we are seeking for H such that $\max_i(d(p, o_i) - r_i) \leq s$ and $\min_j(d(p, o_j) -$

Algorithm 3.3: Target localization

input : A set of sensors $S = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_1(o_2, r_2), \ldots, \mathcal{D}_N(o_N, r_N)\}$, a set of detecting sensors $H = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_1(o_2, r_2), \ldots, \mathcal{D}_k(o_k, r_k)\}$, order-$k$ maximum Voronoi diagram of disks'

$S' = \{\mathcal{D}_1(o_1, r_1 + r_{max}), \mathcal{D}_1(o_2, r_2 + r_{max}), \ldots, \mathcal{D}_N(o_N, r_N + r_{max})\}$

output: The region $\mathcal{R}$ of possible target's location

$r_{\max} \leftarrow \max_{i \in H} r_i$;

for $C_i \in H$ **do**

  $r_i \leftarrow r_{\max} - r_i$;

for $C_j \in S \backslash H$ **do**

  $r_j \leftarrow r_{\max} + r_j$;

Construct order-k region $\mathcal{V}^k(H, S)$;

$H' \leftarrow$ neighboring sets of $\mathcal{V}^k(H, S)$;

Construct $\mathcal{R}$ by inclusion and exclusion of disks in H and $H' \backslash H$;

$r_j) > s,\ \forall i,j : \mathcal{D}_i \in H, \mathcal{D}_j \in S \backslash H$. *Let* $r_{\max} = \max_i r_i\ \forall \mathcal{D}_i \in S$. *The problem is equivalent to searching the set of nearest neighbors to* p *amongst disks* $S' = \{\mathcal{D}_{1'}(o_1, r_{\max} - r_1), \ldots, \mathcal{D}_{N'}(o_N, r_{\max} - r_N)\}$ *that can detect* p. *The main challenge is that the cardinality of* H *is an unknown priori (otherwise, we can just perform a single kNN query for* p*). Instead, we apply the sequential kNN queries and iteratively search for the next nearest disk until the new disk found can no longer detect* p.

The algorithm for likelihood evaluation (Algorithm 3.4) is analogous to Algorithm 3.2 with a stopping criteria to ensure all sensors identified can detect p. *Once* H *is found, likelihood of the measurements can be evaluated by* (3.2) *and* (3.3).

Let K *be the number of sensors that are able to detect the target. From Lemma 3.3, Algorithm 3.4 runs in* $O(\log n + K)$ *on average and* $O(N \log N)$ *in the worst case. The worst case complexity can be improved by "opting" out for the naive approach, which incurs a* $O(N)$ *complexity during the execution of the algorithm.*

3.5.4 Target tracking with location uncertainty and measurement errors

We demonstrate the application of the measurement model developed in the previous section in target tracking defined in Section 3.5.2. We employ the particle filter approach [5] to solve this problem.

The particle filter is a non-parametric implementation of the Bayesian filter, which uses the Bayesian theorem to estimate the posterior probability of system states. Particle filters approximate the posterior distribution by a finite number of samples

Algorithm 3.4: Likelihood evaluation

Input : A set of disks $S = \{\mathcal{D}_1(o_1, r_1), \mathcal{D}_1(o_2, r_2), \ldots, \mathcal{D}_N(o_N, r_N)\}$, order-$k$ max VDs $V^1(S), V^2(S), \ldots, V^{N-1}(S)$; target location p; binary measurement vector $Z = \{z_1, z_2, \ldots, z_N\}$

Output: The likelihood w of sample p for reading Z

$\mathcal{V}(H_1, S) \leftarrow$ *the order-1 max Voronoi region containing* p ;

$H \leftarrow H_1$;

while *The last sensor in* H *is able to detect* p **do**

  $\mathbb{V} \leftarrow$ new vertices of $\mathcal{V}(H_{k-1}, S)$;

  $\mathbb{H} \leftarrow$ disks associated with $\mathbb{V}$, which are not in H_{k-1} ;

  $\mathcal{R} \leftarrow \left\{\mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S)\right\}, \forall \mathcal{D}_i \in \mathbb{H}$;

  $\mathcal{V}(H_k, S) \leftarrow \mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S) \in \mathcal{R} : p \in \mathcal{V}^k(H_{k-1} \bigcup \{\mathcal{D}_i\}, S)$;

  $H \leftarrow H \bigcup \{\mathcal{D}_i\}$;

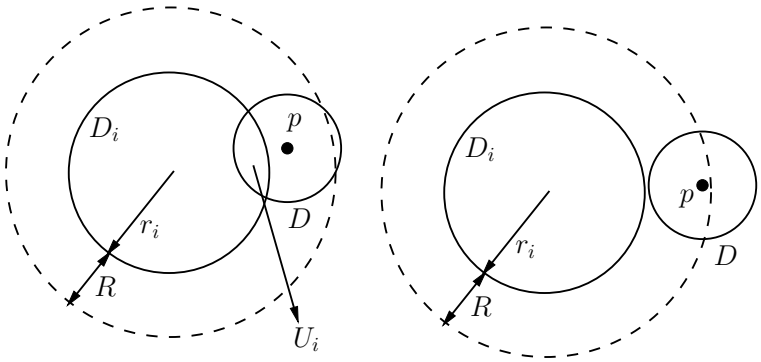
Remove the last element of H;

$w \leftarrow 1$;

$w \leftarrow w \times w_n^m \times \mathcal{P}(z_i|p), \forall z_i \in H$, using (3.2);

$w \leftarrow w \times w_n^m \times \mathcal{P}(z_j|p), \forall z_j \in S \backslash H$, using (3.3);

Return w;



(a) Sensor $\mathcal{D}_i$ is able to detect a target at p.

(b) Sensor $\mathcal{D}_i$ is NOT able to detect a target at p.

Figure 3.8: A sensor with sensing range R and uncertainty radius r_i is possible to detect the target that resides within a disk $(o_i, r_i + R)$ (dashed).

(particles). At time t_n, the system's state is presented by a set of samples randomly drawn from the posterior set of time t_{n-1}. Samples are not chosen with equal probabilities. Instead, they are drawn based on their fitness with the observation at t_n. The more likely a sample generates the observation, the higher probability it is chosen. Denote $\mathbf{X}_n = \left\{X_n^1, X_n^2, \cdots, X_n^M\right\}$ the set of particles at time t_n, where $X_n^m = (x_n^m, y_n^m)$ are the coordinates of the target. Let Z_n and w_n^m, respectively, denote the observation and the importance (defined next) of the m^{th} particle at time t_n. $Z_n = \{z_n^1, z_n^2, \cdots, z_n^N\}$, where z_n^i is the binary value of the i^{th} sensor ($\mathcal{D}_i$) at t_n, i.e., a "1" if the i^{th} sensor detects the target, and "0" otherwise. It has been shown in [71] that:

$$w_n^m = \eta\, \mathcal{P}\left(Z_n \,|X_n\right) = \eta \prod_{i=1}^{M} \mathcal{P}\left(z^i \,|X_n^m\right) \tag{3.6}$$

where η is the normalization factor. The second equality is due to the independence

Algorithm 3.5: Particle filter

Input : The sample set at time t_{n-1}, $\mathbf{X}_{n-1} = \{X_{n-1}^1, X_{n-1}^2, \cdots, X_{n-1}^M\}$; the observation at time t_n, Z_n

Output: The sample set $\mathbf{X}_n = \{X_n^1, X_n^2, \cdots, X_n^M\}$

1 $\mathbf{X}_n \leftarrow \emptyset$;

2 $\mathbf{X}'_n \leftarrow \emptyset$;

3 **for** $i \leftarrow 1$ **to** M **do**

4 $\quad X_n^{'i} \leftarrow$ `MotionModel(`X_{n-1}^i`)` ;

5 $\quad w_n^i \leftarrow$ `MeasurementModel(`$X_n^{'i}$, Z_n`)` from (3.6);

6 $\quad \mathbf{X}'_n \leftarrow \mathbf{X}'_n \bigcup \left\langle X_n^{'i}, w_n^i \right\rangle$

7 **for** $i \leftarrow 1$ **to** M **do**

8 $\quad$ *draw* $X_n^{'i}$ *with probability* $\propto w_n^i$;

9 $\quad \mathbf{X}_n \leftarrow \mathbf{X}_n \bigcup X_n^{'i}$;

10 return $\mathbf{X}_n$;

of sensors' location and observations. The procedure in Algorithm 3.4 can be used to evaluate w_n^m.

The basic particle filter based tracking procedure is given in Algorithm 3.5. To constrain the target's movements, we assume that it can only move within a circle a radius r_t *in between two consecutive observations. Other motion models can be adopted straightforwardly. Line 4 randomizes a position given the constraint. The measurement model (line 5) evaluates the importance of a particle based on Algorithm 3.4.*

3.5.5 Evaluations

We have implemented the proposed algorithms in CGAL [1], a computation geometry library. In this section, we evaluate the performance of target localization and tracking in terms of running time and accuracy. All experiments are conducted in a Core-2 duo 1.7 Ghz workstation.

3.5.5.1 Running time

We evaluate the running time of Algorithm 3.4 for likelihood evaluation. As evident from Section 3.5.4, it is a critical step in target tracking. We randomly select 100000 locations in several deployments with different number of sensors, varying from 5000 to 50000, randomly deployed in areas of 750×500. *The uncertainty radius of sensor locations is uniformly distributed in* $[0, 30]$. *The sensing range is set to be 50. The naive method that scans all sensors is used as a baseline. As shown in Figure 3.9, as the number of sensors increases, the running time of the naive method increases linearly, while that of the proposed method increases much slower. Therefore, the proposed method is more suitable when there is a large number of sensors. Note that by no means is our implementation optimal. So it is expected that further reduction in computation time can be achieved.*

Next, we evaluate the running time of Algorithm 3.4 with respect to the number of sensors detecting the target. In this experiment, we generate 10000 randomized samples in a field of 50000 sensors. We perform kNN queries for k *from 100 to 1000. For each* k, *we execute 50 queries. Figure 3.10 shows a linear increase in the*

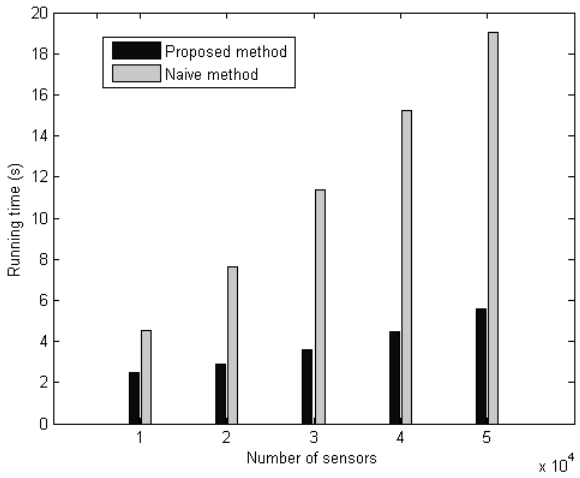


Figure 3.9: Running time of Algorithm 3.4 in comparison with the naive method. X-axis shows the number of sensors. Y-axis shows the execution time (in second).

running time of the proposed algorithm as the number of sensors detecting the target increases. The result agrees with the analysis that the running time is linear with respect to the number of detecting sensors. The speed of increment is relatively slow. In Figure 3.10, when the number of sensors detecting target increases by 10 folds (from 100 to 1000), the running time only increases by 0.4 second.

Next, we evaluate the running time of Algorithm 3.3 to generate the set of sensors for inclusion and exclusion. First, we fix the number of sensors at 1000 and vary k*, the number of sensors detecting targets. The running time is plotted in Figure 3.11a. As shown in the figure, the running time increases according to* $O(k^2 \log k)$ *as proved earlier. Then, we fix* k *at 10 and vary the number of sensors from 500 to 2000. Disk centers are generated with similar density with uncertainties uniformly generated in* $[0, 30]$*. The results are shown in Figure 3.11b. As shown in the figure, the running*

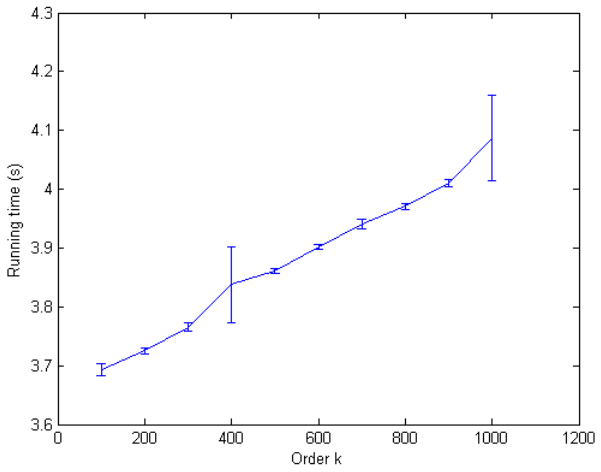


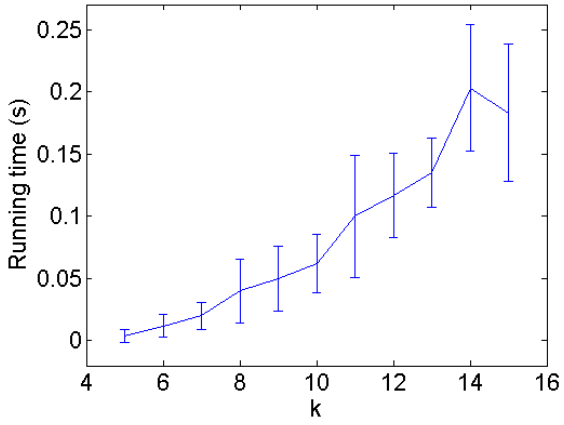
Figure 3.10: Running time of Algorithm 3.4 with respect to the numbers of sensors detecting the target. X-axis shows the numbers of sensors detecting the target. Y-axis shows the running time (in second).

time does not change much as the number of sensors increases. This empirically verifies the claim in Theorem 3.4, which states that the running time of Algorithm 3.3 only depends on the number of sensors that detect the targets.

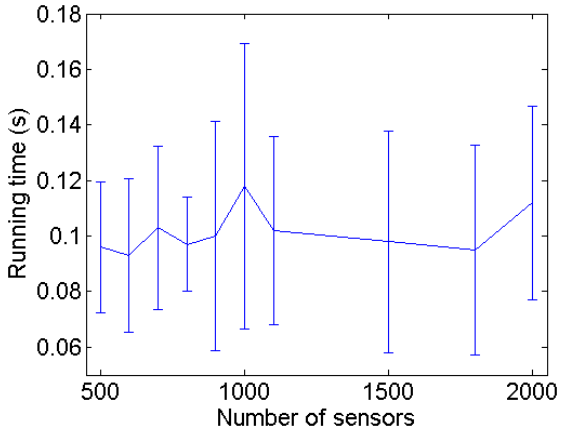
3.5.5.2 Localization and tracking results

In the set of experiments, we examine the impact of sensor location uncertainty on the resolution of localized areas. We randomly deploy 63 sensors in a 200×150 *area with sensing range 50. We increase the sensor location uncertainties from 10 to 45. As shown in Figure 3.12, the size of the localized area increases linearly with the uncertainty radius.*

Finally, we evaluate the performance of the target tracking algorithm. 63 sensors



(a) Fixed number of sensors and various numbers of detecting sensors.



(b) Fixed number of detecting sensors and various numbers of sensors.

Figure 3.11: Running time of Algorithm 3.3 for generating the set of sensors necessary for inclusion and exclusion.

are randomly placed in a 200×150 *area. Each sensor is associated with a location uncertainty uniformly distributed in [0, 20]. Two sensing error probabilities are simulated, namely,* $\epsilon = 0.1$ *and* $\epsilon = 0.15$*. A unique trajectory is calculated from*

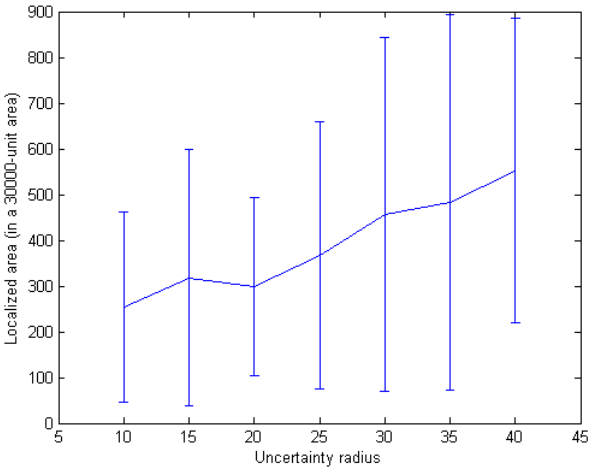
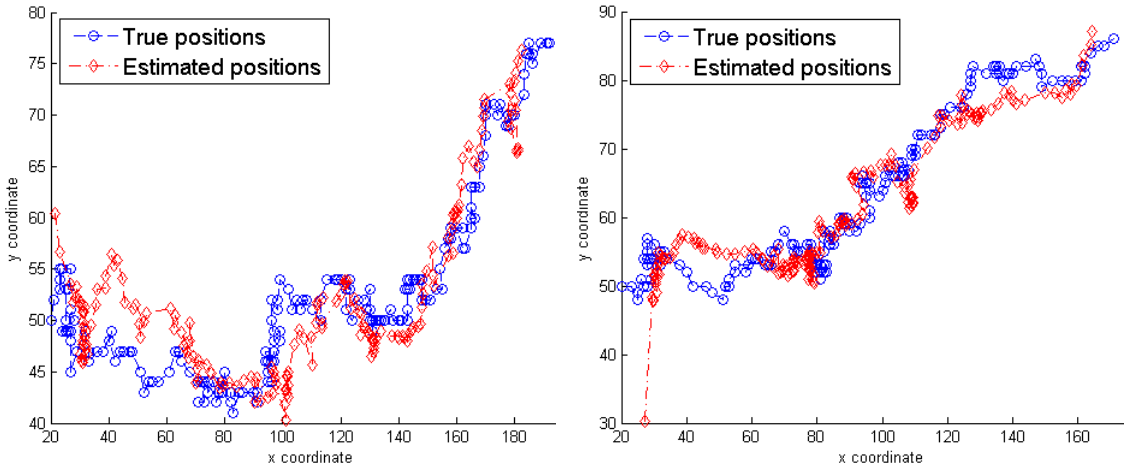


Figure 3.12: The size of localized areas with respect to location uncertainties. X-axis presents the uncertainty radius. Y-axis presents the size of the localized area in the 200×150 region.

the weighted centroid of all samples produced by the particle filter at each time instance. We observe from Figure 3.13 that the computed trajectories (marked by red diamonds) in general match well with the ground truth (marked by blue circles). A higher sensing error probability leads to larger deviation as expected.



(a) $\epsilon = 0.15$ (b) $\epsilon = 0.1$

Figure 3.13: Target tracking with respect to sensing error probability (ϵ's). The diamond chain gives the trajectory estimates. The circle chain represents the true trajectory. XY-axes present the x-y coordinate of the target's location.

Chapter 4

Location Privacy in Participatory Sensing

4.1 Introduction

With the proliferation of mobile devices loaded with rich sensory peripherals, participatory sensing – outsourcing sensing tasks to a large group of mobile users (a crowd) – has gained much attention in a variety of applications including, real-time traffic and road monitoring, reporting spots of oil spill, finding the best biking routes, scoring 3G broadband services, etc. In participatory sensing, a user can both contribute valuable information (data reporting) as well as retrieve (location-dependent) information (query). Privacy is an important issue in data sharing. In participatory sensing, privacy concerns arise from two aspects. The first is in the data reporting

process. It is often desirable to build an understanding/a model of the sensed environment without the precise knowledge of individual's information. Many techniques have been proposed in literature by transforming the data (e.g., adding noise [3], fitting [4], etc.) The second is in the query process, where a user sends location sensitive queries regarding his surroundings (e.g., "where is the closest pub?"). Location privacy mainly concerns with two objectives: hide user locations, and hide user identities, which avoids association of users with their activities (e.g., "who is requesting the nearest pub?"). Our work deals with the latter.

K-anonymity is a measure of privacy first introduced by Sweeney et al. [68] to prevent the disclosure of personal data. A table satisfies K-anonymity if every record in the table is indistinguishable from at least $K-1$ other records with respect to every set of quasi-identifier attributes. In the context of location privacy, the location attribute can be viewed as a quasi-identifier. K-anonymous location privacy thus implies that the user's location is indistinguishable from at least $K-1$ other users. To achieve K-anonymous location privacy, one common approach is to incorporate a trust server, called the anonymizer who is responsible for removing the user's ID and selecting an anonymizing spatial region (ASR) containing the user and at least $K-1$ users in the vicinity (Figure 4.1(b)). Another purpose of ASR is to reduce the commnication cost between the anonymizer and the service provider, and the processing time at service provider side. This process is also called "cloaking" as it constructs a spatial cloak around the user's actual location. The anonymizer forwards the ASR along with the query to the (untrusted) location based service (LBS) (Figure 4.1(c)), which processes the query and returns to the anonymizer a set of

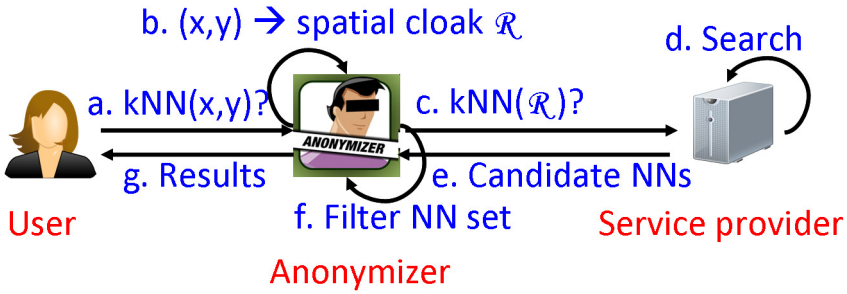


Figure 4.1: Framework for K-anonymous location privacy. kNN stands for the k-nearest neighbor query. NN stands for "nearest neighbor".

candidate point of interests (POIs) (Figure 4.1(e)). The anonymizer removes the false hits and forwards the actual result to the user (Figure 4.1(f)(g)).

As shown in Figure 4.1, in achieving K-anonymous location privacy, it is crucial to devise quality spatial cloaks at the anonymizer and efficient searching algorithms at the LBS. Intuitively, the cloaks produced should be locality preserving – close to the user location, and small in size since both the computational complexity of the search algorithms and the number of POIs returned increases with the size of the cloak. In this paper, we make the following contributions in K-anonymous location privacy for participatory sensing applications:

- **Locality-preserving cloaking:** *We utilize locality-sensitive hashing (LSH) [17] to project the location data to a high-dimension space, which is then partitioned into cells that contain at least K users. LSH has the property that location proximity is preserved during the mapping.*

- **Efficient and flexible search algorithm:** *We devise a search algorithm for*

finding k-nearest POIs of simple polygonal cloaks that takes $O(\log n + Kn + m)$ worst-case running time where m is the number of vertices in the polygonal cloak, n is the number of the POIs, and K is the anonymity level. (In fact, the exact running time is $O(\log n + e + m)$, where $e << O(Kn)$, as shown later.) Contrary to the general belief that complex cloak shapes drastically increase the running time of the k-nearest neighbor (kNN) search, the complexity of our proposed algorithm depends only linearly on m.

The rest of the chapter is organized as follows. In Section 4.3, we introduce the terminology used and the attacker model. The LSH-based cloaking algorithm is described in Section 4.4, and the search algorithm is presented in Section 4.5. In Section 4.6, we present evaluation results. We first introduce the related work for spatial anonymization.

4.2 Related work

Location privacy in participatory sensing campaign is intensively studied in literature ([12, 59, 60, 26]). In the scenario proposed in [37], users have access to a list of data collection sites to and choose the site closest to them. In the work, Kazemi et al. proposed PiRi, a privacy framework that utilizes representative participants for range queries independent of query issuers' location. PiRi assumes that participants can trust one another, and thus may subject to insider attacks.

Privacy in location-based services has drawn much attention in the database and

data mining community in recent years. An excellent survey can be found in [23]. Existing approaches broadly fall into two categories: user-side approaches and approaches that require a trusted server. In the first category, users anonymize their location-based queries by adding noise to the location attributes or generating multiple decoys at different locations. One such approach is called SpaceTwist [78]. In SpaceTwist, starting with a location different from the user's actual location, the nearest neighbors are retrieved incrementally until the query is answered correctly. The uncertainty of the user location is roughly the distance from the initial location to the user's actual location. SpaceTwist requires implementation of incremental k-NN query on the server sides. Furthermore, it does not guarantee K-anonymity if the resulting uncertain area contains less than $K-1$ other users.

With a trusted anonymizer, more sophisticated spatial cloaking mechanisms can be devised. In Casper [49], the anonymizer maintains the locations of the clients using a pyramid data structure, similar to a Quad-tree. Upon reception of a query, the anonymizer first hashes the user's location to the leaf node and then move up the tree if necessary until enough neighbors are included. Hilbert cloaking [35] uses the Hilbert space filling to map 2-D space into 1-D values. These values are then indexed by an annotated B+-tree, which supports efficient search by value or by rank (i.e., position in the 1-D sorted list). The algorithm partitions the 1-D sorted list into groups of K users. Hilbert cloaks though achieving K-anonymity does not always preserve locality, which leads to large cloak size and high server-side complexity. Recognizing that Casper does not provide K-anonymity, Ghinita et al. proposed a framework for implementing reciprocal algorithms using any existing spatial index on the user

locations [24]. Once the anonymous set (AS) is determined, the cloak region can be represented by rectangles, disks, or simply the AS itself.

Specialized (LBS-side) algorithms have been proposed for identifying a candidate set that includes the k nearest neighbors for any location in a convex m-vertex polygon [31]. The authors proposed a sweep-line-based algorithm with $O(mk^2n\log n)$ *worst-case time complexity, which incurs a higher complexity than the proposed Voronoi diagram-based approach. In [35], the authors proposed an algorithm for circular cloaked region with worst-case exponential time complexity. Different from aforementioned work, we propose a search algorithm with any simple polygonal shape cloak with improved running time complexity. The algorithm can be easily extended to handle circular cloaks. Finally, cryptographic approaches have been applied to location privacy, where one-way hash functions are used to encode user and POI locations [52]. Both exact and approximate nearest-neighbor search can be supported at the expense of higher computation complexity.*

4.3 Background

In this section, we introduce necessary terminologies and definitions, and the attacker model.

4.3.1 Attacker model

Similar to [35], we assume an attacker that i) intercepts the ASR, ii) knows the cloaking algorithm used by the anonymizer, and iii) can obtain the up-to-date locations of all users. The first assumption implies that either the LBS is not trusted, or the communication channel between the anonymizer and the LBS is not secure. The second assumption is common in the literature since the data security techniques are typically public. The third assumption is motivated by the fact that users often issue queries from the same locations (home, office), which could be identified through physical observation, triangulation, telephone catalogs, etc.

4.3.2 K-anonymity and reciprocity

We consider N users distributed on a 2-D bounded area $\mathcal{B}$. The set of user locations is denoted by S. The proposed methodologies can be easily extended to higher-dimensional space. We assume queries are one-time (or snapshot queries) such that the attackers cannot utilize historical data to make further inference. Privacy in publishing trajectory data has been considered in [70] and is out of the scope of this paper.

K-anonymity is satisfied if the attacker can identify the user that issues a query with probability not exceeding $1/K$. Reciprocity is introduced by Kalnis [35] as a sufficient condition for K-anonymity as follows:

Definition 4.1. *Consider a user U issuing a query with anonymity degree K, and anonymizing spatial region ASR. ASR satisfies reciprocity if (i) it contains U and at*

least $K-1$ additional users, and (ii) every user in ASalso generates the same ASR for the given K.

Reciprocity necessarily implies a fixed partition of $\mathcal{B}$ such that every partition contains at least K users forming the cloak of the associated users. Though reciprocity is not necessary to ensure K-anonymity, it is easy to verify and has been widely adopted in literature.

4.4 LSH-based cloaking

As discussed in Section 4.3, given a query from location q, the anonymizer needs to construct a spatial cloak that contains q and $K-1$ other user locations. To achieve reciprocity, the anonymizer first partitions all user locations into non-overlapping buckets each containing at least K users. Then, user locations in the bucket containing q are enclosed in a geometric shape that is locality-preserving. In this section, we first show by an example that satisfying both locality-preservation and K-anonymity is not trivial. Then, we propose a LSH-based approach for cloaking.

4.4.1 A naive approach to cloaking

In order-K VDs, each cell corresponds to a set of K sites that are spatially close. Therefore, order-K VD cells appear to be natural candidates for cloaking. Specifically, we first construct an order-K VDs of all user locations S. A cloak can then be formed from a set of sites H of a non-empty cell, which is randomly chosen such that

H contains the query location. Consider the example shown in Figure 4.2. There are 11 user locations in the field. Assume user location p_3 requires 2-anonymity. The order-2 VD of S is given in Figure 4.2. A cell is randomly chosen whose sites contain p_3. Without loss of generality, let the resultant cell be $H = \{p_6, p_3\}$[1]*. H constitutes a cloak for p_3. Unfortunately, this approach does not satisfy K-anonymity if the attacker knows all user locations and the cloaking algorithm. In this case, the probability that the attacker can correctly guess p_3 can be derived using the Bayes formula:*

$$\begin{aligned}\mathbb{P}(p_3|H) &= \frac{\mathbb{P}(H|p_3) \times \mathbb{P}(p_3)}{\mathbb{P}(H)} \\ &= \frac{\mathbb{P}(H|p_3) \times \mathbb{P}(p_3)}{\sum_{p_i \in S} \mathbb{P}(H|p_i) \times \mathbb{P}(p_i)}\end{aligned}$$

The numbers of cells whose sites contain p_3 and p_6 are 4 and 5, respectively. Therefore, $\mathbb{P}(H|p_3) = 1/4$ and $\mathbb{P}(H|p_6) = 1/5$. Additionally, $\mathbb{P}(H|p_i) = 0$ for $p_i \neq p_3, p_6$. We further assume that each user has equal probability to issue the query, and thus $\mathbb{P}(p_3) = 1/11$. Hence,

$$\mathbb{P}(p_3|H) = \frac{\frac{1}{4} \times \frac{1}{11}}{\frac{1}{11} \times (\frac{1}{4} + \frac{1}{5})} > 1/2$$

Clearly, 2-anonymity is violated. The main reason is that different user locations may contribute to different numbers of cells in the order-K VD. Thus, although the approach is locality-preserving, it does not satisfy K-anonymity. This motivates us to seek for a method that is both locality-preserving and K-anonymous.

[1] *For the ease of presentation, we abuse the notation and use H to refer to both the set of sites and the Voronoi cell.*

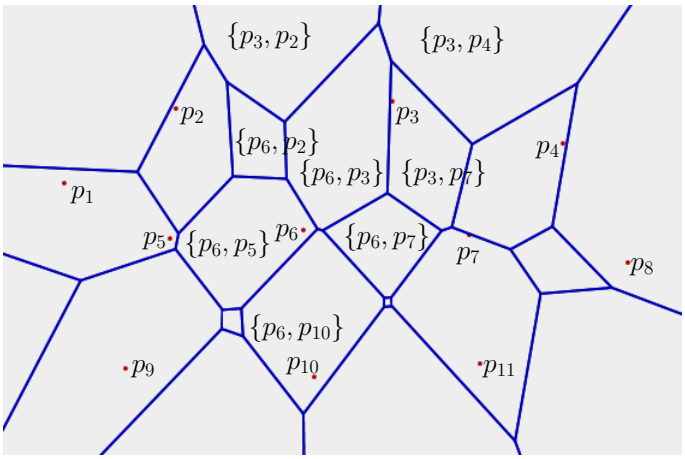


Figure 4.2: The order-2 Voronoi diagram of 11 locations. $\{\cdot, \cdot\}$ shows the locations corresponding to a cell.

4.4.2 Locality-sensitive hashing-based approach to reciprocity

A straightforward approach to reciprocity is to partition the user locations into buckets of adjacent points. For instance, given S, we randomly choose a point p and group it with its $(K-1)$ nearest neighbors. The process continues until all points are assigned to some buckets. This approach has two disadvantages. First, it fragments the dataset and is not locality preserving. Points in the same bucket may be not neighbors in the original dataset, especially for large K's. This is illustrated in Figure 4.3(a). In the example, $K = 4$ and point p is chosen initially. Three neighbors of p then form a cloak, shown in the rectangle. After the four points are removed from the dataset, the cloak of the remaining points is large. In this example, the partition as shown in Figure 4.3(b) is clearly more desirable. Second, the time complexity is high. It is easy to see that the method takes $O(\frac{n^2}{K} \log(\frac{n^2}{K}))$ running time. We show shortly

an approach that achieves desirable partitions with lower time complexity thanks to locality-sensitive hashing (LSH) [17].

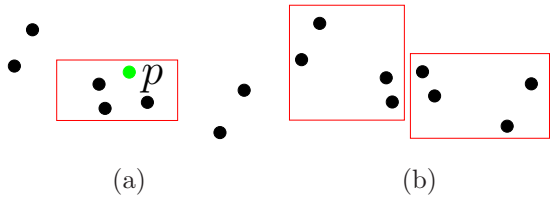


Figure 4.3: Fragmentation in the naive nearest neighbor partition.

LSH hashes the input data so that similar points are mapped to the same buckets with high probability. Formally, for a domain S, a function family $\mathcal{H} = \{h : S \to U\}$ is called (r_1, r_2, p_1, p_2) sensitive with distance measure D if for any $v, q \in S$: if $d(v,q) < r_1$ then $\mathcal{P}_{\mathcal{H}}[h(q) = h(v)] \geq p_1$; if $d(v,q) > r_2$ then $\mathcal{P}_{\mathcal{H}} h(q) = h(v) \leq p_2$, where $p_1 > p_2$, $r_1 < r_2$, and $d(v,q)$ is the distance between v and q in D. There are several LSH families. In this work, we use the LSH family based on p-stable distributions [83] proposed in [17]. The idea of the LSH scheme is as follows. Consider two input vectors v_1 and v_2, and a vector a whose entries are chosen independently from a p-stable distribution $\mathbf{X}$. The distance between their projections on a, $(a \cdot v_1 - a \cdot v_2)$, is distributed as $||v_1 - v_2||_p \mathbf{X}$. By dividing the projected points into equal-width buckets of size r, vector a gives rise to a locality-sensitive hash function, where $h_a(q) = q \cdot a$. However, though similar points are hashed into the same bucket with high probability, the reverse does not hold, i.e., a bucket may contain faraway points. A solution to this problem is to use multiple hash functions. That is, q is hashed by L functions $g_l(q) = \langle h_1(q) \rangle, l = 1, 2, \ldots, L$.[2] *Multiple hash functions lead to a better separation*

[2] *In the original LSH scheme, each hash function maps a data point in $\mathcal{R}^d$ to the $\mathcal{R}^k$. Here we*

of the data points as illustrated in the example in Figure 4.4. The projections of points p_1, p_2, p_3, and p_4 in the plane onto line a are close, while those corresponding to line b are more separate. The use of the two vectors a and b maps p_1 and p_2 into the same bucket, and p_3 and p_4 to another bucket. It has been shown in [17] that given an error rate ε, L can be chosen such that r-near neighbor queries are answered correctly with the error rate lower than ε.

LSH is useful in devising spatial cloaks due to its locality-preserving property. Instead of finding r-near neighbors, the canonical applications of LSH, we wish to partition the data set into groups of at least K elements. Since the partition using a single hash function may contain many distant points, we use L hash functions $g_1, g_2, \ldots, g_L$ instead. The LSH-based partitioning algorithm is summarized in Algorithm 4.1. We first build sorted lists $l_1, \ldots, l_L$ of hashed values of S from the L hash functions. Then, each list is partitioned into buckets each containing K elements (the last one may contain more than K elements). To avoid fragmentation, we always start from the first available point q on l_1. q's $(K-1)$ nearest neighbors are extracted from $U(q)$ the union of the L buckets containing q in the respective lists. Due to the properties of LSH, q's nearest neighbors are in $U(q)$ with high probability, and, moreover, the size of $U(q)$ is not high. In our implementation, the 2-stable Gaussian distribution proposed in [17] is adopted for 2-D location data.

We now analyze Algorithm 4.1's running time. Lines 1 to 3 sort the data set in L lists, which costs $O(L\,n \log n)$, where $n = |S|$ is the cardinality of the dataset. Line 12 requires sorting elements from L buckets, which takes at most $O(L\,k \log Lk)$.

choose $k = 1$.

Algorithm 4.1: LSH-based location partitioning

Data: A set of points S, K-anonymity

Result: T, a partition of S into groups of size K, except for the last one

generate L hash functions, each is a vector whose entries are chosen from a Gaussian distribution;

compute and maintain L sorted lists $\{l_1, \ldots, l_L\}$ of hash values of S;

$T \leftarrow \emptyset$;

while *sorted lists are not empty* **do**

    Partition l_i into buckets of size K, $i = 1, \ldots, L$;

    $q \leftarrow$ the first element of l_1 ;

    $\Omega \leftarrow \emptyset$;

    for $i = 1$ **to** L **do**

        $b \leftarrow$ the bucket containing q in l_i;

        $\Omega \leftarrow \Omega \bigcup b$;

    end

    $NNs \leftarrow q \bigcup (K-1)$ nearest neighbors of q in Ω;

    $T \leftarrow T \bigcup NNs$;

    remove elements of NNs from the L sorted lists ;

end

return T;

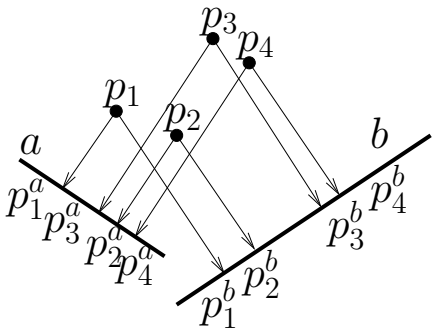


Figure 4.4: Hashing 4 points in the plane with 2 hash functions, a and b.

(Note that the exact number of distinct elements is generally lower). While loop (lines 4 to 14) executes $\lfloor n/k \rfloor$ *times. Therefore, the worst-case running time complexity of Algorithm 4.1 is* $O(L\, n \log n)$*. Since number of distinct elements sorted in line 12 is low, Algorithm 4.1's running time is expected to be inversely proportional to anonymity level* K*.*

User locations produced by Algorithm 4.1 are then used to form spatial cloaks. The search algorithm introduced in Section 4.5 allows simple polygon cloaks as input. Convex polygonal cloaks are popular since they are locality-preserving, and more importantly, the complexity of k*NN queries of convex polygonal cloaks is roughly proportional to the number of edges. However, forming the convex cloak of* k *users takes* $O(K \log K)$ *time, which only needs to be done one time as long as the user locations do not changes. Alternatively, the minimum bounding rectangle can be constructed in* $O(K)$ *time to form the spatial cloaks [72].*

4.5 k-nearest neighbor search for polygonal cloaks

In this section, we first give the necessary and sufficient condition for determining the kNN of a polygonal cloak; and then propose an algorithm. The algorithm can be easily extended to circular cloaks and be omitted due to space limit.

4.5.1 Necessary and sufficient conditions for the kNN set

A spatial region C is said to intersect with a cloak $\mathcal{R}$ if there exists a point p that is interior to both $\mathcal{R}$ and C, and a point p' that is interior to C but exterior to $\mathcal{R}$. In other words, C intersects with $\mathcal{R}$ iff $C \not\subset \mathcal{R}$ and $\mathcal{R} \cap C \neq \emptyset$. C is inside $\mathcal{R}$ if $C \subset \mathcal{R}$. We use the term overlap when C intersects with $\mathcal{R}$ or lies completely inside $\mathcal{R}$.

Given a cloak $\mathcal{R}$, we search for the set $\mathcal{P}$ of POIs that contains the set of k-nearest POIs of any location in the cloak, i.e., $\mathcal{P}$ should be sufficient. Moreover, $\mathcal{P}$ should be necessary, that is, any POI in $\mathcal{P}$ must be in the set of k nearest POIs of some location in the cloak. By the definition of order-k VDs (Section 4.3), the set of sites associated with the Voronoi cells that intersect with $\mathcal{R}$ must be in $\mathcal{P}$. Next, we show formally it is both necessary and sufficient to consider these order-k VD cells.

Lemma 4.1. *Consider a spatial cloak $\mathcal{R}$ of a query point q. Let U be the set of POIs associated with the Voronoi cells of the order-k VD of POIs that intersect with $\mathcal{R}$. U uniquely characterizes the candidate set for the k nearest POIs of q.*

Proof. To show that U is necessary, we prove by contradiction that no POI can be removed from U. Assume $u \in U$ can be removed from U. By the construction of U,

there exists a cell C of sites H such that $u \in H$ and $C \cap \mathcal{R} \neq \emptyset$. Choose any point $p \in C$. Clearly, u is one of the k-nearest neighbor of p, a contradiction.

To show that U is sufficient, consider a POI $u \notin U$ and is one of the k-nearest neighbor of some point $p \in \mathcal{R}$. By the definition of order-k VDs, there exists a cell C corresponding to sites H such that $u \in H$ and $p \in C$. Thus, we have a contradiction. □

4.5.2 Search algorithm

Lemma 4.1 establishes that to determine the kNN of a cloak $\mathcal{R}$, it suffices to identify the order-k cells that overlap with $\mathcal{R}$, and take the union of their corresponding POIs. Next, we first give a straw-man approach that has high computation complexity and then present a more efficient algorithm that utilizes order-1 VDs.

To find the order-k cells overlapping with $\mathcal{R}$, one can start with one such cell, say, C and iteratively explores its neighboring cells that overlap with $\mathcal{R}$. The procedure stops when no such neighboring cells can be found. To find C, we choose an arbitrary point p in $\mathcal{R}$ and query the cell that contains p. Given an order-k VD of n sites, this takes $O(\log n)$. Testing the overlap of a cell with the cloak involves checking the relative position of the cell's edges with respect to the cloak. Let $T(m)$ be the running time of testing whether an edge intersects with $\mathcal{R}$, where m is the number of vertices of $\mathcal{R}$. This procedure takes $O(\log n + e_k \cdot T(m) + s)$, where e_k is the number of Voronoi edges of the order-k cells that overlap with the cloak, and s is the number of POIs returned. Clearly, e_k increases as $\mathcal{R}$ gets larger. In Section 4.6, we implement this

method (called naive search) as a baseline for comparison purposes.

In our proposed approach, we reduce the complexity of the above procedure by considering cells that intersect with $\mathcal{R}$ and cells that are inside $\mathcal{R}$ separately. Furthermore, we show that in the latter case, it suffices to examine order-1 cells, which is generally k times less than the number of order-k cells. To see so, let us consider an example in Figure 4.5, which asks for the 2-nearest POIs of a rectangle cloak (in green). Figures 4.5(a) and (b) show respectively order-2 and order-1 Voronoi diagrams of 12 POIs, $\mathcal{P} = \{p_1, \ldots, p_{12}\}$, which are also the candidate POIs of the cloak. In Figure 4.5a, the POIs given by the cells intersecting with the cloak (darker) is $U_1 = S \backslash \{p_8\}$, while the 11 cells inside $\mathcal{R}$ (white) only contribute one additional POI, namely, p_8. On the other hand, the set of POIs associated with the two order-1 cells inside the cloak is $U_1 = \{p_5, p_8\}$. Obviously, $U_1 \bigcup U_2$ gives all candidate 2-nearest POIs. However, we observe the number of order-1 cells inside the cloak is much smaller (and thus requires less time to identify). We state the results formally as follows:

Lemma 4.2. *Given a simple polygon cloak $\mathcal{R}$ and the order-k VD of a set $\mathcal{P}$ of POIs, $V^k(\mathcal{P})$. Let $I \subset \mathcal{P}$ be the set of sites whose order-1 cells are inside $\mathcal{R}$. The following holds:*

$$\bigcup_{C_j^k \; inside \; \mathcal{R}} H_j^k \subset I \cup \left(\bigcup_{C_i^k \; intersectwith \; \mathcal{R}} H_i^k \right)$$

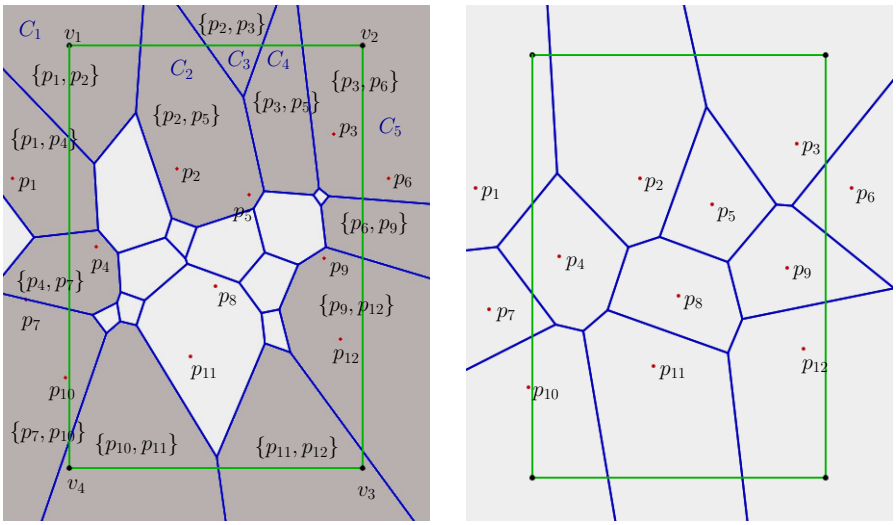
where C_i^k and H_i^k denote the order-k Voronoi cell i in $V^k(S)$, and the set of associated sites, respectively.

Proof. We prove by induction on k. Clearly, the claim is true when $k = 1$. Assume it holds when $k = l$. We prove that it holds when $k = l + 1$. From [43], $V^{l+1}(S)$ can be constructed from $V^l(S)$ by tessellating order-l cells C_i^l's using the sites associated with C_i^l's neighbors. Thus, the Voronoi cell C_i^{l+1} inside $\mathcal{R}$ must be created by tessellating cells C_j^l's inside $\mathcal{R}$ or intersecting with $\mathcal{R}$ with sites corresponding to C_j^l's neighbors, which are either inside $\mathcal{R}$ or intersect with $\mathcal{R}$. Therefore, $\left(\bigcup_{C_j^{l+1}\ inside\ \mathcal{R}} H_j^{l+1}\right) \subset \left(\bigcup_{C_l^m\ inside\ \mathcal{R}} H_l^l \cup \bigcup_{C_i^l\ intersect\ \mathcal{R}} H_i^l\right) \subset \left(I \cup \left(\bigcup_{C_i^m\ intersect\ \mathcal{R}} H_i^l\right)\right)$. The cells of order-$(m+1)$ that intersect with $\mathcal{R}$ are created by tessellating order-m cells that intersect with $\mathcal{R}$ or are inside $\mathcal{R}$. Thus, $\left(\bigcup_{C_i^l\ intersect\ \mathcal{R}} H_i^l\right) \subset \left(\bigcup_{C_i^{l+1}\ intersect\ \mathcal{R}} H_i^{l+1}\right)$. This implies that $\left(\bigcup_{C_j^{l+1}\ inside\ \mathcal{R}} H_j^{l+1}\right) \subset \left(I \cup \left(\bigcup_{C_i^{l+1}\ intersect\ \mathcal{R}} H_i^{l+1}\right)\right)$. □

Therefore, in determining the k-nearest POIs of cloak $\mathcal{R}$, we devise procedures to find POIs associated with order-k cells intersecting with $\mathcal{R}$ and order-1 cells inside the cloak, respectively.

Evaluating candidate k-nearest POIs from cells intersecting with the cloak

First, we identify the cells intersecting with the cloak $\mathcal{R}$ by tracing $\mathcal{R}$'s boundary. W.l.o.g., we assume that vertices $v_1, v_2, \ldots, v_m$ of the cloak, and Voronoi vertices of each cell are in a counter-clockwise order. We start from a vertex v_1 of $\mathcal{R}$ and find the Voronoi cell containing v_1. Let it be C_1. Note that, since Voronoi cells are convex, they intersect with a line segment at most twice. A Voronoi cell that has less than two intersections with a line segment must contain at least one of its endpoints. As a result, we can test if v_2 is in C_1. If yes, we conclude that the line segment $\overline{v_1v_2}$ is in C_1 and continue to the next line segment $\overline{v_2v_3}$. Otherwise, $\overline{v_1v_2}$ must intersect with



(a) Order-2 Voronoi diagram of 12 POIs. Cells intersecting with the cloak is shown darker.

(b) The corresponding order-1 Voronoi diagram of the 12 POIs and the cloak.

Figure 4.5: Illustration of Lemma 4.2: finding 2-nearest POIs. The cloak is presented by the green rectangle $v_1v_2v_3v_4$. $\{\cdot, \cdot\}$ presents the POIs associated with a cell. $C_1, \ldots, C_4$ show four of the order-2 cells intersecting with the cloak.

one unique edge of C_1, which is a bisector of C_1 and one of its neighboring Voronoi cell, say C_2. C_2 and C_1 only differ in one site as given in the following lemma [43]:

Lemma 4.3. *Let H_1 and H_2 be the sets of sites corresponding to two adjacent cells C_1 and C_2 in an order-k Voronoi diagram. Then, $H_1 = H \bigcup \{s_1\}$ and $H_2 = H \bigcup \{s_2\}$. Furthermore, s_1 and s_2 construct the bisector that contains the edge of C_1 and C_2.*

In other words, as we move among neighboring cells intersecting with the cloak, only one POI is added at a time. The above procedure is repeated until the first vertex v_1 is encountered again. Let the resulting set of POIs be U. Consider the example in

Figure 4.5a. Assume we start from cell C_1, POIs p_1 and p_2 are included in U. As we traverse from v_1 to v_2, we move to C_2, which differs from C_1 by p_5. Similarly, C_3 differs from C_2 by p_3, and so on.

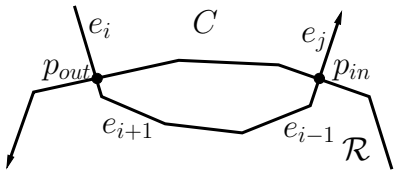


Figure 4.6: Evaluation of the position of edges of cells in B regarding $\mathcal{R}$. C denotes a cell in B.

Evaluating candidate k-nearest POIs from cells inside the cloak *The next step is to retrieve the order-1 cells inside $\mathcal{R}$ and compute the corresponding candidate POIs.*

We discuss the generalized problem of identifying the order-k cells I that are inside the cloak. The idea is to find the cells B intersecting with $\mathcal{R}$, which act as a boundary for the cells inside $\mathcal{R}$. We divide I into those that are adjacent to cells in B (I_1), and the rest, which are not (I_2). The cells in I_1 must share edges with cells in B, and these edges must be inside $\mathcal{R}$. Identifying edges of cells in B that are inside $\mathcal{R}$ can be done as follows. Consider a cell C that intersects with $\mathcal{R}$. As we move along $\mathcal{R}$'s boundary in a counter-clockwise order, we enter C at point p_{in} and exit at point p_{out}. Although C may intersect with $\mathcal{R}$ at more than 2 points, we can always identify such pairs (albeit multiple of them). Now, we observe that, as one moves along C's boundary from p_{out} to p_{in} in the counter-clockwise direction,

the edges of C encountered that do not include p_{out} or p_{in} must be inside $\mathcal{R}$. This allows us to identify the cells of I_1. An illustrative example is given in Figure 4.6. Starting from point p_{out} in edge e_i where the cloak $\mathcal{R}$ exits from C, we trace along C in the counter-clockwise direction (arrow) until we reach point p_{in} on edge e_j where $\mathcal{R}$ enters C. Edges $e_{i+1}, \ldots, e_{j-1}$ are inside $\mathcal{R}$.

To find I_2, we simply iterate through the neighbors of cells that are inside $\mathcal{R}$ until no new inner cells are encountered. Consider the example shown in Figure 4.7. The cloak $\mathcal{R}$ is given by the green polygon. We start at vertex v_1 of $\mathcal{R}$, which lies in cell C_1. Applying the procedure described in the previous section, we can retrieve the shaded cells $C_1, \ldots, C_4$ that intersect with the boundary of $\mathcal{R}$. Next, cells inside $\mathcal{R}$ that are adjacent to cells of B, I_1, are identified, shown in white cells. As shown in the figure, those cells are adjacent to cells in B at the edges that lie completely inside $\mathcal{R}$. Finally, remaining cells inside $\mathcal{R}$ are identified by retrieving neighbors of cells in I_1, shown in red.

We summarize the procedure in Algorithm 4.2.

Complexity analysis *We now analyze the running time of Algorithm 4.2. Let m be the number of $\mathcal{R}$'s vertices. Lines 1 and 3 compute the cells of order-k (B) and order-1 (B') intersecting with $\mathcal{R}$. It first locates the cell containing a cloak's vertex, which costs $O(\log n)$. Then, it iterates through all line segments of the cloak $\mathcal{R}$ and all edges of the order-K and order-1 cells intersecting with $\mathcal{R}$, thus costs $O(e_K + e_1 + m)$, where e_k is the number of edges of order-k cells intersecting with $\mathcal{R}$. Line 4 computes order-1 cells I_2 that are interior to $\mathcal{R}$ and adjacent to B', which iterates all edges*

Algorithm 4.2: Computing the cloak's candidate K nearest POIs.

Data: A Voronoi diagram V, convex polygon cloak $\mathcal{R}$

Result: The set U of K-nearest POIs associated with cells overlapping $\mathcal{R}$

`/* POIs associated with order-`k` cells intersecting with` $\mathcal{R}$ `*/`

$B \leftarrow$ order-K cells intersecting with $\mathcal{R}$;

$U \leftarrow$ POIs associated with B;

`/* POIs associated with order-1 cells inside` $\mathcal{R}$ `*/`

$B' \leftarrow$ order-1 cells intersecting with $\mathcal{R}$;

$curr\_cells \leftarrow$ order-1 cells next to B' that are inside $\mathcal{R}$;

$U' \leftarrow$ POIs associated with $curr\_cells$;

$B' \leftarrow B' \cup curr\_cells$;

while $curr\_cells \neq \emptyset$ **do**

  $curr\_cells \leftarrow curr\_cells$'s neighbors inside $\mathcal{R}$ that are not in B' ;

  $U' \leftarrow$ POIs associated with $curr\_cells$;

  $B' \leftarrow B' \cup curr\_cells$;

end

$U \leftarrow U \cup U'$;

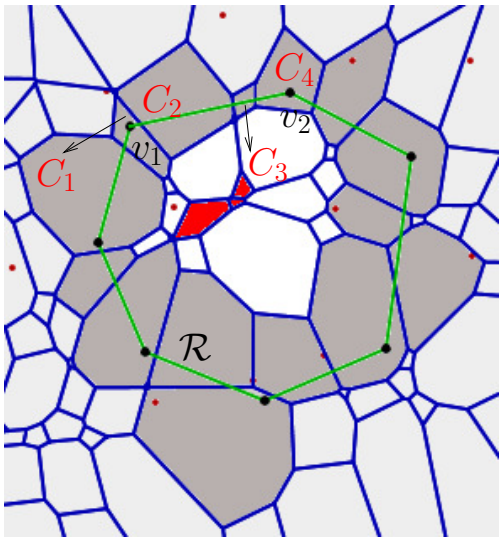


Figure 4.7: Illustration of Algorithm 4.2

of cells in B'. Line 8 computes the other order-1 cells that are inside $\mathcal{R}$, which costs the number of their edges. Let $\mathbf{C}$ be the set of order-k cells intersecting with $\mathcal{R}$ and order-1 cells overlapping $\mathcal{R}$ Computing U (lines 2, 5, 9) takes $O(s + c)$, where s is the number of POIs returned by the algorithm, and $c = |\mathbf{C}|$. (Note that $(e_K + e_1)$ is smaller than the number of $\mathbf{C}$'s edges). Therefore, Algorithm 4.2 costs $O(\log n + s + e + m)$ running time, where e is the number of $\mathbf{C}$'s edges. Since the number of edges in an order-K Voronoi diagram is $O(Kn)$ ([43]), e is bounded by $O(Kn)$. In practice, $e << O(Kn)$ due to the fact that the cloak is small and usually convex.

4.6 Evaluation

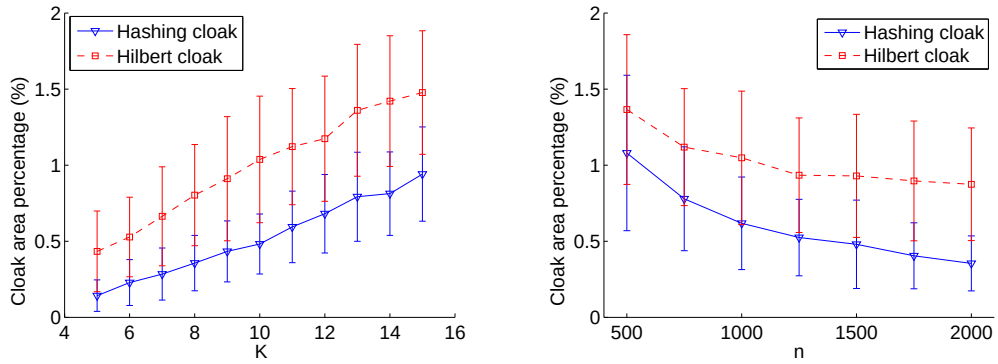
Results of hashing-based cloaking and POI search algorithms have been implemented in CGAL [1], a computational geometry library. All simulations run on a Core2 Duo 1.7Ghz Linux workstation.

4.6.1 Cloaking

In the first set of experiments, we compare the performance of the proposed cloaking method with the Hilbert cloaking method. In the simulations, n user locations are randomly placed in a 1000×1000 *area.*

Figure 4.8 compares the performance of the two methods with respective to the level of K-anonymity where the number of users where n fixed at 1000 and K fixed at 10, respectively. The y-axis gives the size of the cloak as the percentage of the size of the area. Also shown in the figures are the error bars corresponding to the maximum and minimum cloak sizes. As can be observed in Figures 4.8a and 4.8b, the LSH-based approach is significantly better than the Hilbert curve-based method. As K increases, the cloak size increases roughly linearly in both methods. With more users, it is expected that the cloak size decreases linearly. This trend is more prominent with the proposed LSH-based method.

The hashing-based method is efficient computationally. Figure 4.9 plots the running time of the hashing-based cloaking method when the number of users varies from 1000 to 10000. As shown in the figure, the hashing-based method's running



(a) The number of hash functions L is 20. n is fixed at 1000. K varies from 5 to 15.

(b) The number of hash functions L is 10. K is fixed at 10. n varies from 500 to 2000.

Figure 4.8: Performance of Hilbert curve-based method and hashing-based method on various levels of K-anonymity and number of users (n).

time is low even with a large number of users. Somehow, counter-intuitively, the running time of the algorithm decreases as the anonymity level K increases since it is inversely proportional to the anonymity level as indicate in Section 4.4.

To evaluate the impact of the number of hash functions used, we randomly generate 1000 locations, and issue 10-nearest POI and 20-nearest POI queries. The number of hash functions, L, varies from 2 to 45. As shown in Figure 4.10, the higher the number of hash functions, the smaller the cloak areas produced. However, the cloak areas slightly decrease as the number of hash functions increases from 10 to 45. Similar to the results in Figure 4.8a, higher K implies large cloaks.

We compare the resultant candidate POI set of the proposed search algorithm corresponding to hashing and Hilbert cloaks. We vary anonymity level K from 5 to 17 and make queries with 500 and 1000 POIs. As shown in Figure 4.11, the number

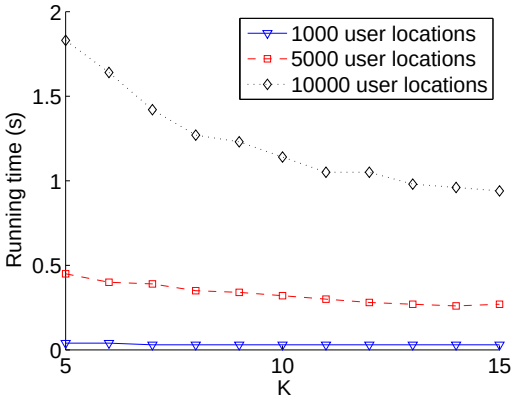


Figure 4.9: Running time of hashing-based cloaking method. The number of hash functions L is 20.

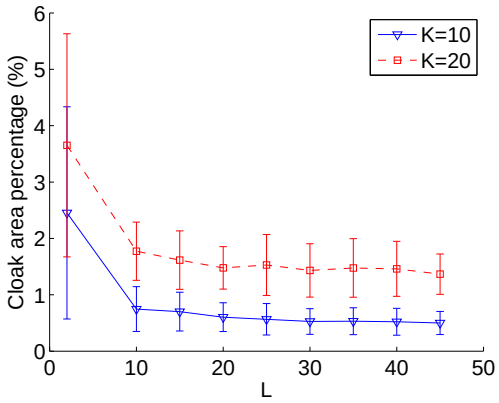


Figure 4.10: Hashing-based cloaking performance with different number of hash functions.

of POIs corresponding to LSB-based cloaks is smaller than that of Hilbert cloaks. Moreover, the difference between the two methods increases as the number of POIs increases from 500 to 1000.

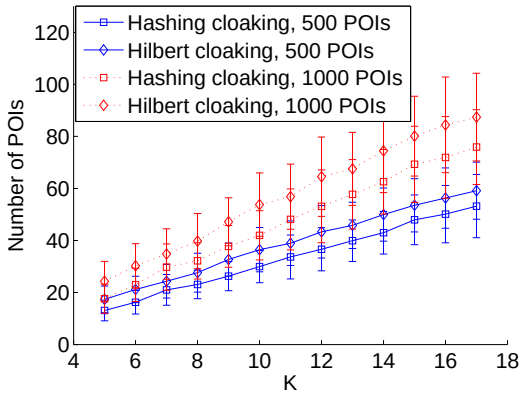


Figure 4.11: Number of candidate POIs corresponding to Hilbert and hashing cloaks.

4.6.2 kNN search

Next, we evaluate the performance of the proposed kNN search algorithm. In the simulations, 1000 POIs are randomly placed in a 1000×1000 *area. We compare the running time with different K-anonymity (also K nearest POIs) and cloak areas (as the percentage of the total area). The results are shown in Figure 4.12. As seen in the figure, the running time is negligible. The running time increases only moderately as K or the size of the cloak area increases, which is sharp contrast with the results in [35].*

Figure 4.13 shows the number of candidate k-nearest POIs. We evaluate the number of candidate POIs with different cloak areas and values of k. As shown in

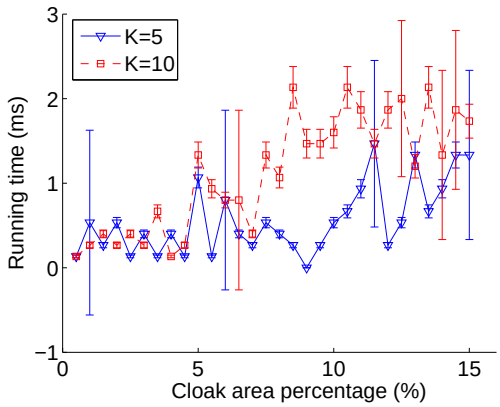


Figure 4.12: Running time of the proposed search algorithm.

the figure, the number of candidate k-nearest POIs increases approximately linearly as the cloak area increases since POIs are evenly distributed.

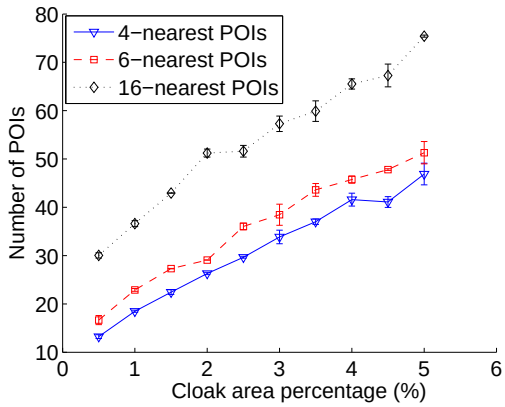


Figure 4.13: Number of candidate k-nearest POIs.

Lastly, we compare the complexity of the proposed algorithm with the naive search algorithm that scans order-k Voronoi cells. We vary the cloak area and plot the number of cells processed. As shown in Figure 4.14, the number of cells processed in the proposed method is much lower. Furthermore, as the cloak area increases, the

number of cells processed by our method increases much slower than that in the naive search. Figure 4.14 corroborates the low running time of the proposed algorithm in Figure 4.12.

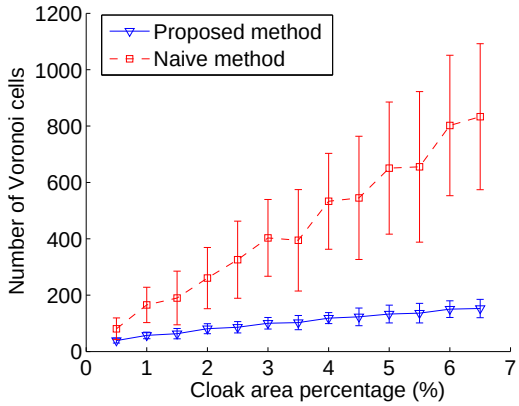
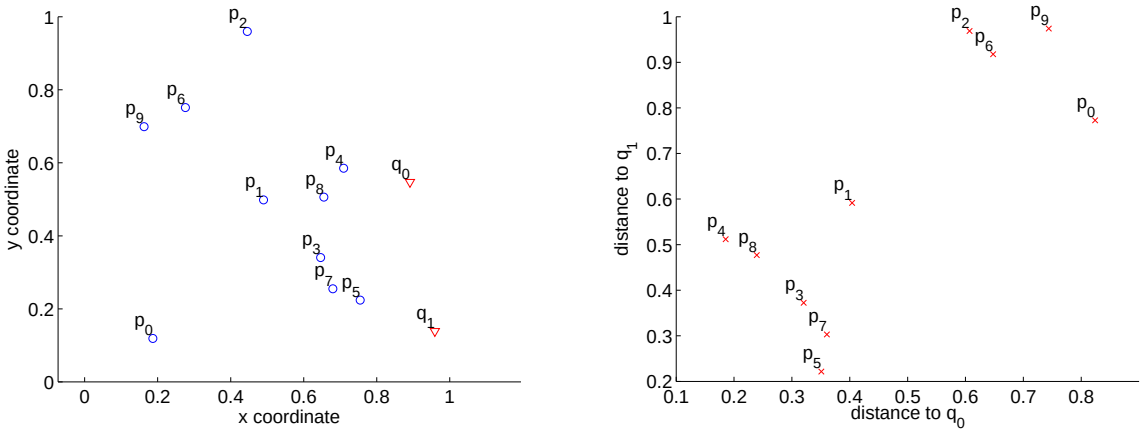


Figure 4.14: Number of cells processed by the proposed method VS. the naive (only considers order-k cells).

Chapter 5

Spatial Skyline Query With Uncertainty

In the previous chapter, we discuss the framework for kNN queries that concerns user privacy. To obtain user privacy, we construct cloak for the user location and make the kNN query for the cloak. Cloaking is also be helpful in various of scenarios, where a spatial query may involve in several cloaks. Queries may involve in several cloaks. For example, find the hotels that are close to a set of users, whose locations are given as circular cloaks. In this chapter, we discuss spatial skyline queries under user location uncertainty. Furthermore, we also take measurement errors into account and discuss the spatial skyline queries with margins.



(a) (b)

Figure 5.1: A spatial skyline query example

5.1 Introduction

In recent years, there has been much interest in processing spatial skyline queries. Given a set of points of interest (POIs), the spatial skyline query for a set of locations returns the POIs that are close to all locations. Consider the following example:

Example 5.1. *A student is looking for an apartment. She prefers the apartments that are close to both Galleria mall and the University of Houston. Figure 5.1 illustrates 10 apartment locations,* $p_0, \ldots, p_9$*, and the locations of Galleria mall and the University of Houston, i.e,* q_0 *and* q_1*, respectively. The distances from apartments* p_i*'s to* q_0 *and* q_1 *are shown in Figure 5.1b.*

Galleria mall and the University of Houston are referred to as query locations, and apartments as POIs. Apartment p_i *is not considered if there is another apartment,* p_j*, that is closer to both* q_1 *and* q_2 *than* p_i*. In this case,* p_i *is said to be dominated*

by p_j, i.e., p_j is closer than p_i w.r.t. any query location. A skyline query returns the POIs, call skyline pointss which are not dominated by any other POI. In this example, p_3, p_4, p_5, p_8 are skyline points.

The distance from p_i to query location q_j is an attribute of p_i. Hence, each POI has n attributes corresponding to n query locations. When distances from query locations to POIs are provided, the skyline query can be done by traditional methods from the database community [41, 69, 14]. However, these approaches are not efficient in the spatial context due to two reasons. First, computing all distances from query locations to POIs is expensive. Second, the distances between query locations and POIs cannot be determined exactly in many scenarios, particularly when uncertainty is introduced to query locations. Third, the domination relationship can be defined differently in different contexts. Consider the following examples.

Example 5.2. *A group of users is looking for a hotel that is close to all members. Due to privacy concern, user locations are given as circular cloaks. Hotel p_i dominates hotel p_j w.r.t. a user q_u if p_i is closer than p_j to any location of q_u in the cloak. p_i dominates p_j if p_i dominates p_j w.r.t. all user locations.*

In Example 5.2, the domination between two POIs w.r.t. a query location still can be evaluated while the distances from POIs to query locations are not given. Clearly, p_i dominates p_j if all query locations lie in the halfplane divided by bisector $b_{i,j}$ of the two POIs that contains p_i. This is illustrated in Figure 5.2. In this example, p_1 dominates p_2 but does not dominate p_3 w.r.t. query locations q_1 and q_2.

Example 5.3. *The scenario is similar to Example 5.1 with different domination*

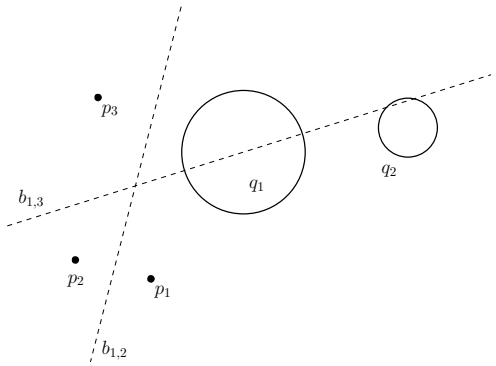


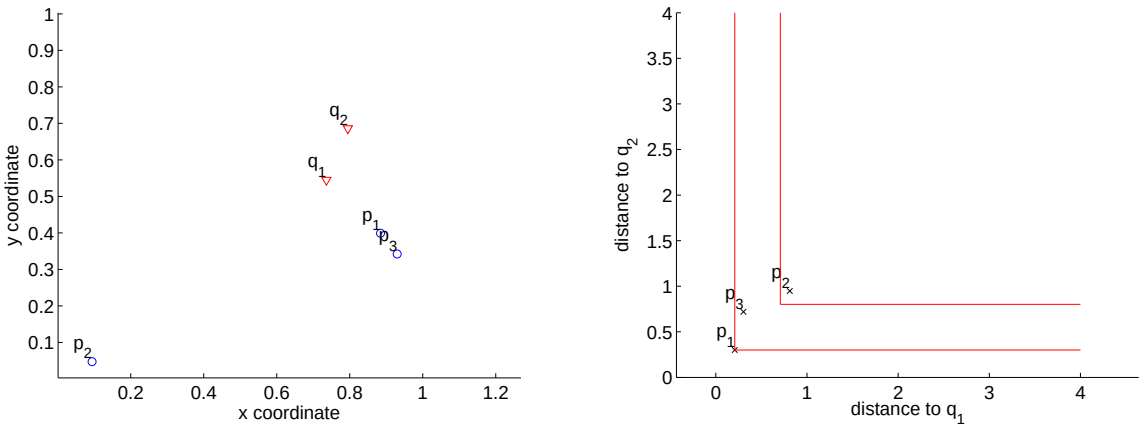
Figure 5.2: Domination relationship with query location uncertainties.

relationship. Apartment p_i dominates apartment p_j w.r.t. q_u if the distance from p_i to q_u is smaller than the distance from p_j to q_u by a pre-defined threshold.

Example 5.3 provides a domination relationship that take measurement errors into account. POI p_i dominates p_j w.r.t. query location q_v if the distance from p_i to q_u is smaller than the distance from p_j to q_u by a margin. Figure 5.3 shows an example of the domination relationship with margins. p_3 is not dominated by p_1 when the margins for both query locations are set at 0.5. In this chapter, we develop techniques to solve the skyline query problems in the two examples.

5.2 Related work

Spatial skyline query gains much interest in recent years. Most work is from the database community. Branch-and-Bound Skyline [56] and the Block Nested Loop [11] algorithms are among the early efforts. Both approachs are for general skyline computation but they can also solve the spatial skyline queries. Their work, however,



(a) 3 POIs (p_1, p_2, p_3) and 2 query locations (q_1, q_2) (b) Distances from POIs to query locations

Figure 5.3: Domination relationship with margins. Red lines show the margin of POI p_1.

does not incorporate knowledge of the geometry into the problems.

Much work in the area of spatial databases concern spatial skyline queries. Papadias et al. [57] proposed efficient algorithms to nd the point with minimum aggregate distance to query points. Huang and Jensen [32] studied the problem of finding the points of interest which are not dominated with respect to two attributes: their network distance to a single query location q and the detour distance from q's predefined route through the road network.

Sharifzadeh and Shahabi are among the first to tackle the spatial skyline query problem using geometric approaches [63]. They proposed a Voronoi diagram based algorithm for spatial skyline queries. Son et al. discovered a flaw in [63] and proposed an improved algorithm in [66], which is also based on the Voronoi diagram. Very

recently, Lin et al. proposed a new type of spatial skyline query [45], where they defined the distance from a POI to a type of query locations as the smallest distances to the elements of the same type. For example, the distance from an apartment to the bus stations is defined as the distance to the nearest bus station.

Spatial skyline queries with different distance metrics have also been investigated. In [67], Son et al. proposed spatial skyline queries with Manhattan distance. [18] tackled spatial skyline queries in road networks. [28] considered the direction of POIs w.r.t. query locations for the domination.

Different from these work, we proposed a new type of spatial skyline queries, which deals with query location uncertainty and flexible domination relationship.

5.3 Problem definition

In the following two problems, we denote $P = \{p_0, p_1, \ldots, p_{n-1}\}$ *to be the set of POIs where each POI* p_i *is located at* $(p_i.x, p_i.y)$*. Let* Q *be the set of query locations. We define the skyline set* $S = \{s_0, s_1, \ldots, s_m\}$ *of* P *w.r.t.* Q *to be the set of POIs such that* s_i *is not dominated by any POI in* P*. We define the two problems based on different domination relationships.*

Problem 5.1. *Spatial skyline queries with location uncertainty. Let* $Q = \{q_0, q_1, \ldots, q_{k-1}\}$ *be the set of query locations where* q_i *is given as a disk* $\mathcal{D}_i(o_i, r_i)$ *centered at* o_i *with radius* r_i*. We say* p_i *dominates* p_j *w.r.t.* q_u *iff* $d(p_i, p_u) < d(p_j, p_u) \forall p_u \in \mathcal{D}_u$*, where* $d(p_i, p_u)$ *is the Euclidean distance between points* p_i *and* p_u*.* p_i *is defined to dominate*

p_j iff p_i dominates p_j w.r.t. every query location in Q. Compute the skyline set of P.

Problem 5.2. *Fuzzy skyline queries. Let $Q = \{q_0, q_1, \ldots, q_{k-1}\}$ be the set of query locations where q_i is located at point $(q_i.x, q_i.y)$. Let $\delta_i > 0$ be a set of margins. p_i is not dominated by p_j w.r.t. q_u if $d(p_i, p_u) \leq d(p_j, p_u) - \delta_i$. p_i is said to dominate p_j w.r.t. q_u iff $d(p_i, p_u) < d(p_j, p_u) - \delta_i$. p_i dominates p_j if p_i is not dominated by p_j w.r.t. any query location, and p_i dominates p_j in some query location. Compute the skyline set of P.*

5.4 Solution approaches

In this section, we provide the solutions for problems raised in the previous section.

5.4.1 Spatial skyline queries with location uncertainty

Consider two POIs p_i and p_j. From the problem definition, p_i dominates p_j iff all query locations Q belong to the half-plane divided by the bisector b_{ij} of p_i and p_j that contains p_i. Therefore, testing the domination of p_i and p_j is equivalent to determining if b_{ij} intersects the convex hull of disks in Q, $CH(Q)$. If b_{ij} intersects $CH(Q)$, p_i and p_j do not dominate each other. Otherwise, p_i or p_j dominates the other depending on the position of $CH(Q)$ w.r.t. b_{ij}. We first make the following observation.

Claim 5.1. *Let $d_i = d(p_i, o_i)$, i.e., the distance from p_i to the center of query location*

q_i. Let $L = \{d_0, d_1, \ldots, d_{n-1}\}$ be the sorted list of distances $\{d_i$'s$\}$ in ascending order. The POI corresponding to d_0 is a skyline point. The POI corresponding to d_i in L can only be dominated by the skyline points corresponding to d_j where $j < i$.

Proof. W.l.o.g., let p_i be the POI corresponding to d_i. Since $d_0 = \min_{j=0}^{n-1} d_j$, no POI dominates p_0 w.r.t. q_0. Thus p_0 is a skyline point. Furthermore, $d_i = \min_{j=i+1}^{n-1} d_j$, no POIs corresponding to $d_{i+1}, \ldots, d_{n-1}$ dominate p_i. It remains to prove that it is sufficient to test dominations of p_i with only skyline points p_j, $j < i$. We show that if p_i is dominated by a non-skyline point p_l, where $l < i$, then p_i must be dominated by a skyline point p_j where $j < i$. In fact, since p_i is dominated by p_l, then $d(p_i, p_v) < d(p_l, p_v) \forall p_v \in q_u, \forall q_u \in Q$. Furthermore, since p_l is not a skyline points, it is dominated by some skyline point p_j where $j < l$. Therefore, $d(p_i, p_v) < d(p_l, p_v) < d(p_j, p_v) \forall p_v \in q_u, \forall q_u \in Q$. Thus, p_i is dominated by p_j. The claim holds. □

Based on this observation, we devise an algorithm to compute spatial skyline queries. It generalizes of the algorithm proposed in [66], which works for point query locations. The algorithm works by incrementally searching skyline points with increasing distances to the center of some query location. A POI p_i is a skyline point iff it is not dominated by all skyline points that are discovered thus far. Details are given in Algorithm 5.1. We first analyze the complexity of the domination test and the expected number of convex disks.

Domination test *Line 7 in Algorithm 5.1 tests if skyline point $S[j]$ dominates the POI p_i corresponding to $L[i]$. Testing if a skyline point p_j dominates POI p_i is determined by checking if the bisector b_{ij} of p_i and p_j intersects the convex hull $CH(Q)$ of Q. This can be done in $O(n \log n)$ time complexity. The idea is as follows. We first find the extreme points of $CH(Q)$ in the direction $\overrightarrow{b'_{ij}}$, where b'_{ij} is the line perpendicular to b_{ij} [54]. Project vertices of $CH(Q)$ onto b'_{ij}. The extreme points of a convex hull w.r.t. $\overrightarrow{b'_{ij}}$ are defined as the points corresponding to the rightmost and leftmost projected points on b'_{ij}. If $CH(Q)$'s vertices are ordered, e.g., in counter-clockwise, finding the highest point of a convex hull is done using binary search as follows. Start with an edge ba of $CH(Q)$ where a is next to b in counter-clockwise direction. If ba does not contain the highest point, then the highest point must be in chain ab. Let c be a vertex between a and b. We search for the highest point in chain ac or cb. We define the direction $\overrightarrow{i}$ of vertex i of $CH(Q)$ as the counter-clockwise direction of the tangent line of $CH(Q)$ at i. We define $\overrightarrow{i}$ is up if the angle between $\overrightarrow{i}$ and $\overrightarrow{b'_{ij}}$ is $\pi/2$ or less. Otherwise, $\overrightarrow{i}$ is down. We consider the case where $\overrightarrow{a}$ is up (Figure 5.4). If $\overrightarrow{a}$ is down, then we look for the highest point in chain ac. If $\overrightarrow{c}$ is up, we consider the position of c and a w.r.t. $\overrightarrow{b'_{ij}}$. If c is above a, chain cb is searched for the highest point. Otherwise, chain ac is considered. The same argument applies when $\overrightarrow{a}$ is down. The lowest point of $CH(Q)$ can be found by the similar procedure. Since the chain size is reduced by a half in each process, finding the extreme points of a convex hull is done in $O(n \log n)$ time. b_{ij} intersects $CH(Q)$ iff the extreme points of $CH(Q)$ w.r.t $\overrightarrow{b'_{ij}}$ lie in different half-planes divided by b_{ij}.*

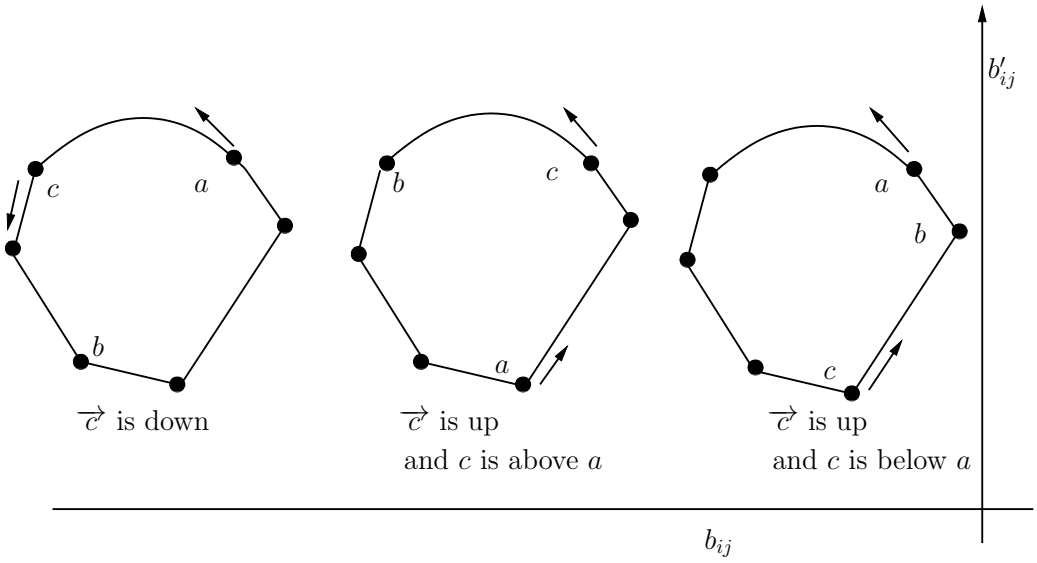


Figure 5.4: Determine intersection of a convex hull and a line.

Expected number of convex disks *It is shown in [76] that the expected number of convex points of a set of n points uniformly distributed in a square is* $O(\log n)$. *We extend the result to show that the expected number of convex disks of a set of disks uniformly distributed in a square is also* $O(\log n)$ *in the following lemma.*

Lemma 5.1. *Given a set of n disks independently distributed in a square, the expected number of convex disks is* $O(\log n)$.

Proof. Let $D = \{\mathcal{D}_0, \ldots, \mathcal{D}_{n-1}\}$ be the set of n disks. We break the hull into four parts the upper-left, upper-right, lower-left, and lower-right hulls and show that each has $O(\log n)$ points. It is sufficient to consider the upper-right hull. Let R_i be the smallest axis-parallel square enclosing disk $\mathcal{D}_i$, and v_i be the upper-right corner point of R_i. Consider the convex hull of v_i's, CH'. Clearly, the necessary condition of $\mathcal{D}_i$ being the convex disk is v_i is the convex point of CH'. Since each disk may have

at most two tangent points on the convex hull of $\mathcal{D}$, the expected number of convex disks of $\mathcal{D}$ is $O(\log n)$. □

We now analyze the running time of Algorithm 5.1. Constructing the convex hull of n disks takes $O(n \log n)$ time by [58]. Line 2 takes $O(n \log n)$ time. Since each POI is tested against skyline points, the time for evaluating a POI is $O(m \log |CH(Q)|)$, where m is the number of skyline points. Hence, lines 4-10 take $O(nm \log k)$ time. Thus the worst case running time of Algorithm 5.1 is $O(nm \log |CH(Q)| + n \log n)$. Since the expected number of convex disks of a set of k disks is $O(\log k)$, the expected running time of Algorithm is $O(nm \log \log k + n \log n)$

Algorithm 5.1: Computing the spatial skyline set.

input *: A set of POIs $P = \{p_0, p_1, \ldots, p_{n-1}\}$, a set of query locations $Q = \{q_0, \ldots, q_{k-1}\}$*

output*: The skyline set S of P*

```
1  compute the convex hull of Q;
2  L ← sort P according to ascending distances o_i;
3  S ← L[0];
4  for i = 0 to n − 1 do
5      is_dominated ← false;
6      for j = 0 to |S| − 1 do
7          is_dominated ← dominationTest(S[j], L[i]);
8      end
9      if !is_dominated then S ← S ⋃ L[i];
10 end
```

5.4.2 Fuzzy skyline queries

In this section, we provide the solutions for Problem 5.2. In particular, we analyze the case when all margins δ_i's are equal, and then, proposed a solution for the general case.

From the problem definition, p_i dominate p_j w.r.t. query location q_u iff $d(p_i, q_u) + \delta_u < d(p_j, q_u)$. In other words, p_i dominates p_j w.r.t. query location q_u iff the maximum distance from p_i to $\mathcal{D}_u$ is smaller than the distance from p_j to q_u, where $\mathcal{D}_u(q_u, \delta_u)$ is the disk centered at q_u with radius δ_u. This means that q_u belongs to the half-plane divided by the bisector of p_j and the disk $D(p_i, \delta_u)$ that contains p_i. Clearly, p_i lies in the convex half-plane. We derive a similar observation as in Claim 5.1.

Claim 5.2. *Let $\mathcal{D} = \{\mathcal{D}_0(q_0, \delta_0), \ldots, \mathcal{D}_{k-1}(q_{k-1}, \delta_{k-1}\}$ be the set of disks associated with query locations Q. Let d_i be the distance from p_i to q_0 and d'_i be the maximum distances from p_i to $\mathcal{D}_0$. Let $L = \{d_0, d'_0, d_1, d'_1, \ldots, d_{n-1}, d'_{n-1}\}$. Sort L in ascending order. Let $L[i]$ be the first maximum distance. The POIs associated with $L[j]$, where $0 \leq j < i$, are skyline points. POI p_i associated with $L[i]$ can only be dominated by POIs whose maximum distances appear before $L[i]$ in L.*

Proof. For the convenience of presentation, we assume that elements in L are distinct. Since $L[i]$ is the first maximum distance in L, $\forall$ POI $p_j, j < i$, $\nexists$ POI p_j such that $d'_l < d_j$, where p_l is the POI associated with the element that appears before $L[i]$ in L. Therefore, p_l is not dominated by any POI, and hence, is a skyline point. Similarly, those POIs whose maximum distances appear after $L[i]$ in L cannot dominate the POI p_i associated with $L[i]$ w.r.t. to q_0, and hence, cannot dominate p_i. Furthermore, if p_i

is dominated by non-skyline point p_j whose maximum distance d'_j appears before p_i in L, then $d_i < d'_l$ where d'_l is the first maximum distance in L. Thus, p_i is dominated by p_l, a skyline point. This completes the proof.

□

To this point, we develop the algorithms for the two cases, i.e., fuzzy skyline queries with equal margins and un-equal margins.

Equal margins *We consider the case that query locations have the same margin δ. Let i be the index of the first maximum distance in L. From Claim 5.2, POIs corresponding to $L[j], j < i$ are skyline points. We iterate through the POIs p_j corresponding to the minimum distances $L[j], j > i$. From Claim 5.2, we only perform domination tests of p_j and the skyline points corresponding to the indices appearing before j in L. Details are given in Algorithm 5.2.*

Algorithm 5.2: Computing the spatial skyline set of margin .

input : *Set of POIs* $P = \{p_0, p_1, \ldots, p_{n-1}\}$, *set of query locations* $Q = \{q_0, \ldots, q_{k-1}\}$, *the common margin* δ

output: *The skyline set* S *of* P

1 $CH(Q) \leftarrow$ *the convex hull of* Q;

`/* min. and max. distances from POIs to query location` q_0 `*/`

2 $L = \{d_0, d'_0, d_1, d'_1, \ldots, d_{n-1}, d'_{n-1}\}$;

3 $L \leftarrow$ *sort* L *in ascending order;*

4 *init* $\leftarrow$ *the index of the first maximum distance in* L;

5 **for** $i = 0$ **to** *init* **do**

6 $\quad S \leftarrow S \bigcup$ *the POI corresponding to* $L[i]$;

7 **end**

8 **for** $i = init + 1$ **to** $n - 1$ **do**

9 $\quad$ *is_dominated* $\leftarrow$ *false;*

10 $\quad$ **for** $j = 0$ **to** $|S| - 1$ **do**

11 $\quad\quad$ *is_dominated* $\leftarrow$ `dominationTest(`$S[j]$, $L[i]$, $CH(Q)$`);`

12 $\quad$ **end**

13 $\quad$ **if** !*is_dominated* **then** $S \leftarrow S \bigcup L[i]$;

14 **end**

The domination test (Line 11) can be done by checking if the bisector of p_j *and disk* $D(p_i, \delta)$ *intersects the convex hull of* Q. *Specifically,*

Lemma 5.2. *Given a set of query locations* $Q = \{q_0, q_1, \ldots, q_{k-1}\}$, *a margin* δ, *and a skyline point* p_i *resulted from steps 6 and 13. Let* b_{ij} *be the bisector of* p_j *and disk*

$D(p_i, \delta)$. The domination test in line 11 can be done by checking the intersection of b_{ij} and $CH(Q)$. If b_{ij} intersects $CH(Q)$, the convex hull of Q, p_i does not dominate p_j. Otherwise, p_i dominates p_j.

Proof. Since skyline point p_i is determined beforehand, $d'_i < d_j$. Thus, q_0 lies in the half-plane containing p_i. Therefore, if b_{ij} does not intersect $CH(Q)$, Q must lie in the half-plane containing p_i. Thus, the maximum distances from p_i to query locations in Q are smaller than the corresponding distances from p_j. Hence, p_i dominates p_j. Furthermore, if b_{ij} intersects $CH(Q)$, there must be some q_u lying in the half-plane containing p_j, thus, $d(p_i, q_u) + \delta < d(p_j, q_u)$. Hence p_i does not dominate p_j. This completes the proof. □

Clearly, the domination test in Algorithm 5.2 takes $O(|CH(Q)|)$ running time. By the similar argument in the analysis of the previous problem, the Algorithm 5.2 takes $O(nm|CH(Q)| + n \log n)$ running time.

Generalization to different margins *Algorithm 5.2 still applies to the case of different margins. However, when the margins of query locations are different, the domination test is more complex since each query location requires a different bisector. This leads to a $O(nmk)$ running time complexity for the domination tests. In fact, the number of skyline points are much smaller than the number of POIs, it is reasonable to spend extra time to pre-process the data for each skyline point so that the domination tests can be done faster.*

Details are given in Algorithm 5.3. The idea is as follows. Given a skyline

point p_i, let $D = \{\mathcal{D}_i(q_i, d(q_i, p_i) + \delta_i)$'s$\}$ be the set of disks associated with query locations. Clearly, p_i dominates a POI p_j if p_j is not inside the unions of disks in D. The union of k disks can be constructed by the algorithms proposed in [7, 33], which take $O(k \log k)$ running time. The complexity of the union boundary is $O(k)$ [38] and querying the position of p_j w.r.t. a union of disks can be done in $O(\log k)$ [40, 46]. Therefore, constructing the unions of disks for m skyline points in line 9 takes $O(mk \log k)$. Evaluating POIs takes $O(nm \log k)$. Thus, the running time is $O(mk \log k + nm \log k + n \log n)$.

Algorithm 5.3: Computing the spatial skyline set.

input : *Set of POIs* $P = \{p_0, p_1, \ldots, p_{n-1}\}$, *set of query locations* $Q = \{q_0, \ldots, q_{k-1}\}$, *set of margins* $\Delta = \{\delta_i\text{'s}\}$

output: *The skyline set* S *of* P

1 $CH(Q) \leftarrow$ *the convex hull of* Q;

2 $L = \{d_0, d'_0, d_1, d'_1, \ldots, d_{n-1}, d'_{n-1}\}$;

3 $L \leftarrow$ *sort* L *in ascending order;*

4 $S \leftarrow \emptyset$;

5 $U \leftarrow \emptyset$;

6 $init \leftarrow$ *the index of the first maximum distance in* L;

7 **for** $i = 0$ **to** $init$ **do**

8 $\quad S \leftarrow S \bigcup$ *the POI corresponding to* $L[i]$;

9 $\quad U \leftarrow U \bigcup$ `disksUnion(`Q, $L[i]$, Δ`)`;

10 **end**

11 **for** $i = init + 1$ **to** $n - 1$ **do**

12 $\quad is\_dominated \leftarrow false$;

13 $\quad$ **for** $j = 0$ **to** $|S| - 1$ **do** $is\_dominated \leftarrow$ `isOutside(`$L[i]$, $U[j]$`)`;

14 $\quad$ **if** $!is\_dominated$ **then**

15 $\quad\quad S \leftarrow S \bigcup L[i]$;

16 $\quad\quad U \leftarrow U \bigcup$ `disksUnion(`Q, $L[i]$, Δ`)`;

17 $\quad$ **end**

18 **end**

Chapter 6

Conclusion

6.1 Summary of Contributions

Our work concerns both the theoretical foundation for sensor network analysis and its applications. On the theory side, we developed several algorithms to construct high order Voronoi-based diagrams, which is of independent interest in the computational geometry community. We proposed the algorithm to construct the high order maximum Voronoi diagram of disks. On the application side, we proposed algorithms for a number of applications in wireless sensor networks based on high order Voronoi diagrams. Particularly, we focus on analysis of sensor location uncertainties that concerns worst case operation of the system, and privacy in participatory sensing.

High order maximum Voronoi diagram of disks

We present the procedures for high order maximum Voronoi diagram update when disks change their sizes. This leads to various results. First, a distributed implementation-friendly algorithm is devised for high order max VD of disks in the plane construction. Second, an $O(kn \log n)$ algorithm or site insertion and deletion in order-k max VDs is proposed. The high order maximum Voronoi diagram is the building block for analyzing sensor networks with sensor location uncertainties, which are modeled as disks.

Sensor location uncertainty in network coverage and target localization and tracking

We investigate the algorithms for dealing with sensor location uncertainty in two tasks including network coverage and target localization and tracking.

In network coverage, we devised geometrical-based methods to compute the minimum sensing range to ensure robust k point and area coverage in presence of sensor location uncertainty. We consider generic settings of arbitrary placements, heterogeneous uncertainty areas, and polygonal area boundaries. Our proposed solution is mainly based on the concept of order-k maximum Voronoi diagrams of disks.

In target localization and tracking, we developed efficient algorithms to evaluate the likelihood of noisy sensor readings and kNN queries, which served as building blocks for target localization and tracking under sensor location uncertainties. The

proposed algorithms can be straightforwardly extended in several directions including i) heterogeneous sensing ranges, ii) uncertainties in sensing ranges, and iii) target localization with noisy measurements (where a stochastic search approach can be adopted in conjunction with the likelihood evaluation algorithm).

Privacy in participatory sensing

We proposed computationally efficient algorithms for constructing spatial cloaks and kNN search of POIs with the dual objectives of locality preservation and K-anonymity. We showed through simulations that the algorithms achieve superior performance with moderate time complexity are scale well with large input size. The proposed algorithms are central to protect identity privacy in participatory sensing applications. Cloaking motivates various spatial queries with consideration of location uncertainty.

Uncertainty in spatial skyline queries

We extend uncertainties into spatial skyline queries and deal with various domination relationships. We proposed geometric algorithms for two spatial skyline queries applications, i.e., to cope with user location uncertainties, and to allow flexibilities in domination relationships. In the former case, we query locations are given as circular cloaks. The worst case domination is defined, i.e., a POI dominates another POI w.r.t. a query location if all locations in the corresponding cloak are closer to the former POI than to the latter. In the latter case, the domination relationship

is defined to allow a margin. Specifically, a POI dominates another POI if the corresponding distances to query locations of the former are smaller than those of the latter with some threshold. This work opens up many interesting problems in spatial skyline queries with different domination relationships.

6.2 Future work

Spatial skyline queries with presence of uncertainties exposes several interesting problems. In Chapter 5, we discussed user location uncertainties and domination uncertainty. In fact, objects are not only points in practical systems, and thus, lead to different distance metrics. Furthermore, domination relationships vary abundantly. These are the main factors that make spatial skyline queries much more challenging. In the scope of our future work, we envision the spatial skyline queries in several contexts with different distance metrics and/or domination relationships. This also exposes the necessity of revising the conditions of skyline points.

Spatial skyline queries with heterogeneous distance metrics and domination relationships

Different POI and/or query location shapes define different distance metrics. We first extend the problem defined in Chapter 5, which deals with circular query locations. Both minimum and maximum distances are used to evaluate the domination relationship.

Example 6.1. *A group of users is looking for a hotel that is close to all members. Due to privacy reason, user locations are given as circular cloaks. Hotel* p_i *dominates hotel* p_j *w.r.t. a user* q_u *if the maximum distance from* p_i *to* q_u *is smaller than the minimum distance from* p_j *to* q_u*.* p_i *dominates* p_j *if* p_i *dominates* p_j *w.r.t. all user locations.*

Example 6.2. *Parks with multiple entrances play as query locations. The distance from an apartment (a POI) to a park is measured as the distance to the closest entrance. To make it easy for analysis, we assume that a park is modeled as a convex polygon, and the distance from an apartment to a park is measured as the smallest distance to the polygon. Given a set of convex polygon query locations and a set of POIs, find the skyline points.*

In the first example, the domination relationship can be evaluated although the distance from a POI to a query location is not concretely defined. In the second one, although the distance from a POI to a query location is defined, a solution that is better than the naive is not trivial. In both examples, the challenge is to find a solution that does not require calculation of all distances.

Domination test

What are the conditions to make a POI skyline point? By definition, a POI that is not dominated by any other POI is a skyline point. However, this requires an $O(n)$ *tests. In Chapter 5, we shown the applications where only dominations test against skyline points are sufficient to determine if a POI is a skyline point. In the*

on-going work, we are looking for the algorithms that can further prune the number of domination tests.

Bibliography

[1] Cgal, *Computational Geometry Algorithms Library. http://www.cgal.org.*

[2] *A. Acquisti and J. Grossklags. Uncertainty, ambiguity and privacy. In* Fourth Workshop On the Economics Of Information Security, (WEIS05) 2005*, pages 2–3, 2005.*

[3] *D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In* PODS '01: Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 247–255, New York, NY, USA, 2001. ACM.*

[4] *H. Ahmadi, N. Pham, R. Ganti, T. Abdelzaher, S. Nath, and J. Han. Privacy-aware regression modeling of participatory sensing data. In* Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 99–112, New York, NY, USA, Nov. 2010. ACM.*

[5] *M. S. Arulampalam, S. Maskell, and N. Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking.* IEEE Transactions on Signal Processing*, 50:174–188, 2002.*

[6] *F. Aurenhammer. Power diagrams: properties, algorithms and applications.* SIAM J. Comput.*, 16:78–96, February 1987.*

[7] *F. Aurenhammer, T. U. Graz, R. Klein, F. Hagen, and P. I. Vi. Voronoi diagrams. In* Handbook of Computational Geometry*, pages 201–290. Elsevier Science Publishers B.V. North-Holland, 199.*

[8] *J. Bachrach and C. Taylor.* Localization in Sensor Networks*, pages 277–310. John Wiley & Sons, Inc., 2005.*

[9] *A. Bar-Noy, T. Brown, M. Johnson, and O. Liu. Covering with inexactly placed sensors. In* FWCG*, 2007.*

[10] *A. R. Beresford and F. Stajano. Location privacy in pervasive computing.* IEEE Pervasive Computing*, 2(1):46–55, Jan. 2003.*

[11] *S. Börzsönyi, D. Kossmann, and K. Stocker. The skyline operator. In* Proceedings of the 17th International Conference on Data Engineering*, pages 421–430, Washington, DC, USA, 2001. IEEE Computer Society.*

[12] *J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. In* In: Workshop on World-Sensor-Web (WSW06): Mobile Device Centric Sensor Networks and Applications*, pages 117–134, 2006.*

[13] *S. Capkun, M. Hamdi, and J.-P. Hubaux. Gps-free positioning in mobile ad hoc networks. In* Cluster Computing*, pages 3481–3490, 2001.*

[14] *C.-Y. Chan, H. V. Jagadish, K.-L. Tan, A. K. H. Tung, and Z. Zhang. On high dimensional skylines. In* Proceedings of the 10th International Conference on Advances in Database Technology*, EDBT'06, pages 478–495, Berlin, Heidelberg, 2006. Springer-Verlag.*

[15] *Y. Chiang and R. Tamassia. Dynamic algorithms in computational geometry. Technical report, Providence, RI, USA, 1991.*

[16] *C.-Y. Chow and M. F. Mokbel. Trajectory privacy in location-based services and data publication.* SIGKDD Explor. Newsl.*, 13(1):19–29, Aug. 2011.*

[17] *M. Datar and P. Indyk. Locality-sensitive hashing scheme based on p-stable distributions. In* SCG 04: Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pages 253–262. ACM Press, 2004.*

[18] *K. Deng, X. Zhou, and H. Tao. Multi-source skyline query processing in road networks. volume 0, pages 796–805, Los Alamitos, CA, USA, 2007. IEEE Computer Society.*

[19] *Q. Fang, J. Gao, and L. J. Guibas. Locating and bypassing holes in sensor networks.* Mob. Netw. Appl.*, 11(2), 2006.*

[20] *Q. Fang, J. Gao, L. J. Guibas, V. Silva, and L. Zhang. Glider: Gradient landmark-based distributed routing for sensor networks. In* Proc. of the 24th Conference of the IEEE Communication Society (INFOCOM*, pages 339–350, 2005.*

[21] S. Funke. Guaranteed-delivery geographic routing under uncertain node locations. In In: Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM07) (2007) K.M. Lillis, S.V. Pemmaraju, and I.A. Pirwani*, 2007.*

[22] S. Funke and C. Klein. Hole detection or: "how much geometry hides in connectivity?". In Proceedings of the Twenty-second Annual Symposium on Computational Geometry*, pages 377–385, 2006.*

[23] G. Ghinita. Private queries and trajectory anonymization: a dual perspective on location privacy. Trans. Data Privacy*, 2:3–19, April 2009.*

[24] G. Ghinita, K. Zhao, D. Papadias, and P. Kalnis. A reciprocal framework for spatial k-anonymity. Inf. Syst.*, 35:299–314, May 2010.*

[25] R. Ghrist and A. Muhammad. Coverage and hole-detection in sensor networks via homology. In Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004.*

[26] P. Gilbert, L. P. Cox, J. Jung, and D. Wetherall. Toward trustworthy mobile sensing. In Proceedings of the Eleventh Workshop on Mobile Computing Systems and Applications*, HotMobile '10, pages 31–36, New York, NY, USA, 2010. ACM.*

[27] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys '03, pages 31–42, New York, NY, USA, 2003. ACM.*

[28] X. Guo, Y. Ishikawa, and Y. Gao. Direction-based spatial skylines. In Proceedings of the Ninth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '10, pages 73–80, New York, NY, USA, 2010. ACM.*

[29] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. A. Stankovic, and T. Abdelzaher. Achieving real-time target tracking usingwireless sensor networks. In RTAS '06: Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 37–48, Washington, DC, USA, 2006. IEEE Computer Society.*

[30] M. Hefeeda and H. Ahmadi. A probabilistic coverage protocol for wireless sensor networks. In Proceedings of the IEEE International Conference on Network

Protocols, ICNP 2007, October 16-19, 2007, Beijing, China*, pages 41–50. IEEE, 2007.*

[31] H. Hu and D. L. Lee. Range nearest-neighbor query. IEEE Trans. on Knowl. and Data Eng.*, 18:78–91, January 2006.*

[32] X. Huang and C. S. Jensen. In-route skyline querying for location-based services. In Proc. of the Int. Workshop on Web and Wireless Geographical Information Systems (W2GIS), Goyang, Korea*, 2004.*

[33] H. Imai, M. Iri, and K. Murota. Voronoi diagram in the laguerre geometry and its applications. SIAM J. Comput.*, 14(1):93–105, 1985.*

[34] M. P. Johnson, D. Sarioz, A. Bar-Noy, T. Brown, D. Verma, and C. W. Wu. More is more: The benefits of denser sensor deployment. In INFOCOM*, pages 379–387, 2009.*

[35] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. IEEE Trans. on Knowl. and Data Eng.*, 19:1719–1733, December 2007.*

[36] M. I. Karavelas and M. Yvinec. Dynamic additively weighted voronoi diagrams in 2d. In Proceedings of the 10th Annual European Symposium on Algorithms*, ESA '02, pages 586–598, London, UK, UK, 2002. Springer-Verlag.*

[37] L. Kazemi and C. Shahabi. Towards preserving privacy in participatory sensing. In 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 328–331. IEEE, Mar. 2011.*

[38] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles. Discrete Comput. Geom.*, 1(1):59–71, Apr. 1986.*

[39] W. Kim, K. Mechitov, J.-Y. Choi, and S. Ham. On target tracking with binary proximity sensors. In Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.*

[40] D. G. Kirkpatrick. Optimal search in planar subdivisions. Technical report, Vancouver, BC, Canada, Canada, 1981.

[41] D. Kossmann, F. Ramsak, and S. Rost. Shooting stars in the sky: an online algorithm for skyline queries. In Proceedings of the 28th International Conference on Very Large Databases*, VLDB '02, pages 275–286. VLDB Endowment, 2002.*

[42] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. J. Mach. Learn. Res.*, 9:235–284, June 2008.*

[43] D.-T. Lee. On k-nearest neighbor voronoi diagrams in the plane. IEEE Trans. Comput.*, 31(6):478–487, 1982.*

[44] X. Li, P. Wan, , and O. Frieder. Coverage in wireless ad hoc sensor networks. IEEE Transactions on Computers*, 52(6), 2003.*

[45] Q. Lin, Y. Zhang, W. Zhang, and A. Li. General spatial skyline operator. In Proceedings of the 17th International Conference on Database Systems for Advanced Applications - Volume Part I*, DASFAA'12, pages 494–508, Berlin, Heidelberg, 2012. Springer-Verlag.*

[46] R. J. Lipton and R. E. Tarjan. Applications of a planar separator theorem. In Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 162–170, Washington, DC, USA, 1977. IEEE Computer Society.*

[47] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. Sensor networks for emergency response: Challenges and opportunities. IEEE Pervasive Computing*, 3(4):16–23, Oct. 2004.*

[48] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In INFOCOM*, 2001.*

[49] C. C. W. A. MF Mokbel. The new casper: A privacy-aware location-based database server. 2007 IEEE 23rd International Conference on Data Engineering*, pages 1499–1500, 2007.*

[50] P. Mohan, V. N. Padmanabhan, and R. Ramjee. Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 323–336, New York, NY, USA, 2008. ACM.*

[51] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In SenSys '04: Proceedings of the

2nd International Conference on Embedded Networked Sensor Systems, *pages 50–61, 2004.*

[52] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011. *The Internet Society, 2011.*

[53] E. Nilsson and K. Stlen. Ad hoc networks and mobile devices in emergency response a perfect match? In J. Zheng, D. Simplot-Ryl, and V. Leung, editors, Ad Hoc Networks, *volume 49 of* Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, *pages 17–33. Springer Berlin Heidelberg, 2010.*

[54] J. O'Rourke. Computational Geometry in C. *Cambridge University Press, New York, NY, USA, 2nd edition, 1998.*

[55] M. H. Overmars. Range searching in a set of line segments. In Proceedings of the First Annual Symposium on Computational Geometry, *SCG '85, pages 177–185, New York, NY, USA, 1985. ACM.*

[56] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. ACM Trans. Database Syst., *30(1):41–82, Mar. 2005.*

[57] D. Papadias, Y. Tao, K. Mouratidis, and C. K. Hui. Aggregate nearest neighbor queries in spatial databases. ACM Trans. Database Syst., *30(2):529–576, June 2005.*

[58] D. Rappaport. A convex hull algorithm for discs, and applications. Comput. Geom. Theory Appl., *1(3):171–187, Mar. 1992.*

[59] S. Reddy, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. A framework for data quality and feedback in participatory sensing. In Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, *SenSys '07, pages 417–418, New York, NY, USA, 2007. ACM.*

[60] S. Reddy, D. Estrin, and M. Srivastava. Recruitment framework for participatory sensing data collections. In P. Floréen, A. Krüger, and M. Spasojevic, editors, Proceedings of the 8th International Conference on Pervasive Computing, *pages 138–155, Berlin, Heidelberg, May 2010. Springer Berlin Heidelberg.*

[61] J. Sember and W. Evans. Guaranteed voronoi diagrams of uncertain sites. In Proceedings of the 20th Annual Canadian Conference on Computational Geometry, Montreal, Canada, August 13-15, 2008, *2008.*

[62] M. I. Shamos and D. Hoey. Geometric intersection problems. In Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, SFCS '76, pages 208–215, Washington, DC, USA, 1976. IEEE Computer Society.*

[63] M. Sharifzadeh and C. Shahabi. The spatial skyline queries. In Proceedings of the 32nd international conference on Very large data bases*, VLDB '06, pages 751–762. VLDB Endowment, 2006.*

[64] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri. Target tracking with binary proximity sensors. ACM Trans. Sen. Netw.*, 5(4):1–33, 2009.*

[65] J. Singh, U. Madhow, R. Kumar, S. Suri, and R. Cagley. Tracking multiple targets using binary proximity sensors. In Proceedings of the 6th International Conference on Information Processing in Sensor Networks, IPSN*, pages 529–538, New York, NY, USA, 2007. ACM.*

[66] W. Son, M.-W. Lee, H.-K. Ahn, and S.-W. Hwang. Spatial skyline queries: An efficient geometric algorithm. In Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases*, SSTD '09, pages 247–264, Berlin, Heidelberg, 2009. Springer-Verlag.*

[67] W. Son, S. won Hwang, and H.-K. Ahn. Mssq: manhattan spatial skyline queries. In Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases*, SSTD'11, pages 313–329, Berlin, Heidelberg, 2011. Springer-Verlag.*

[68] L. Sweeney. k-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10:557–570, Oct. 2002.*

[69] K.-L. Tan, P.-K. Eng, and B. C. Ooi. Efficient progressive skyline computation. In Proceedings of the 27th International Conference on Very Large Data Bases*, VLDB '01, pages 301–310, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.*

[70] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In Proceedings of the the Ninth International Conference on Mobile Data Management*, pages 65–72, Washington, DC, USA, 2008. IEEE Computer Society.*

[71] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.*

[72] G. T. Toussaint. Solving geometric problems with the rotating calipers. In Proceedings of IEEE MELECON*, 1983.*

[73] K. Vu and R. Zheng. Robust coverage under uncertainty in wireless sensor networks. In 2011 Proceedings IEEE INFOCOM (INFOCOM 2011)*, pages 2015–2023, Shanghai, P.R. China, 4 2011.*

[74] Q. Wang, R. Zheng, A. Tirumala, X. Liu, and L. Sha. Lightning: A hard real-time, fast, and lightweight low-end wireless sensor election protocol for acoustic event localization. IEEE Trans. Mob. Comput.*, 7(5):570–584, 2008.*

[75] Y. Wang, J. Gao, and J. S. Mitchell. Boundary recognition in sensor networks by topological methods. In MobiCom '06: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, pages 122–133, 2006.*

[76] R. Wenger. Randomized quick hull. Algorithmica*, 17, 1995.*

[77] J. White, C. Thompson, H. Turner, B. Dougherty, and D. C. Schmidt. Wreckwatch: Automatic traffic accident detection and notification with smartphones. Mob. Netw. Appl.*, 16(3):285–303, June 2011.*

[78] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 366–375, Washington, DC, USA, 2008. IEEE Computer Society.*

[79] M. L. Yiu, E. Lo, and D. Yung. Authentication of moving kNN queries. In Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 565–576, April 2011.*

[80] T.-H. You, W.-C. Peng, and W.-C. Lee. Protecting moving trajectories with dummies. In Proceedings of the 2007 International Conference on Mobile Data Management*, MDM '07, pages 278–282, Washington, DC, USA, 2007. IEEE Computer Society.*

[81] H. Zhang and J. C. Hou. Maintaining sensing coverage and connectivity in large sensor networks. Ad Hoc & Sensor Wireless Networks*, 1(1-2), 2005.*

[82] R. Zheng, K. Vu, A. Pendharkar, and G. Song. Obstacle discovery in distributed actuator and sensor networks. ACM Trans. Sen. Netw.*, 7(3):22:1–22:24, Oct. 2010.*

[83] *V. M. Zolotarev. One-dimensional stable distributions. In* Translations of Mathematical Monographs*, volume 65. American Mathematical Society, 1986.*

[84] *Y. Zou and K. Chakrabarty. Sensor deployment and target localization in distributed sensor networks.* ACM Trans. Embed. Comput. Syst.*, 3(1):61–91, Feb. 2004.*

[85] *Y. Zou and K. Chakrabarty. Uncertainty-aware and coverage-oriented deployment for sensor networks.* J. Parallel Distrib. Comput.*, 64(7):788–798, July 2004.*