

M.Sc. ENGG. THESIS

**Algorithms for r -Gathering and
 r -Gather Clustering Problems**

By
Shareef Ahmed

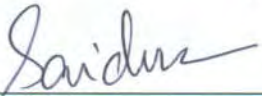

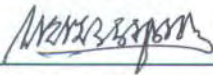
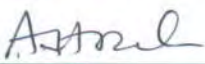

Submitted to
Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science & Engineering

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology(BUET)
Dhaka-1000, Bangladesh

March, 2019

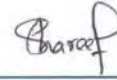
The thesis titled “Algorithms for r -gathering and r -gather clustering problems”, submitted by Shareef Ahmed, Roll No. 0416052006 P, Session April 2016, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents. Examination held on March 28, 2019.

Board of Examiners

1. 
Dr. Md. Saidur Rahman
Professor
Department of CSE
BUET, Dhaka 1000.
Chairman
(Supervisor)
2. 
Dr. Md. Mostofa Akbar
Professor & Head
Department of CSE
BUET, Dhaka 1000.
Member
(Ex-officio)
3. 
Dr. M. Kaykobad
Professor
Department of CSE
BUET, Dhaka 1000.
Member
4. 
Dr. Atif Hasan Rahman
Assistant Professor
Department of CSE
BUET, Dhaka 1000.
Member
5. 
Dr. Md. Rezaul Karim
Professor
Department of CSE
University of Dhaka, Dhaka 1000.
Member
(External)

Candidate's Declaration

This is to certify that the work presented in this thesis entitled "**Algorithms for r -gathering and r -gather clustering problems**" is the outcome of the investigation carried out by me under the supervision of Professor Dr. Md. Saidur Rahman in the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology (BUET), Dhaka. It is also declared that neither this thesis nor any part thereof has been submitted or is being currently submitted anywhere else for the award of any degree or diploma.



Shareef Ahmed
Candidate

Contents

<i>Board of Examiners</i>	ii
<i>Candidate's Declaration</i>	iii
Acknowledgements	viii
Abstract	ix
1 Introduction	1
1.1 r -Gathering Problem	3
1.2 r -Gather Clustering Problem	4
1.3 Application of r -Gathering	5
1.4 Application of r -Gather Clustering	6
1.5 Literature Review	7
1.6 Scope of The Thesis	8
1.7 Contributions	9
1.8 Thesis Organization	9
2 Preliminaries	10
2.1 Basic Terminology	10
2.1.1 Graphs	10
2.1.2 Directed and Undirected Graphs	11
2.1.3 Paths and Cycles	12
2.1.4 Complete Graphs	12

2.1.5	Trees	12
2.1.6	Star graphs	13
2.2	Algorithms and Complexity	13
2.2.1	Big- O Notation	14
2.2.2	Polynomial Algorithms	15
2.2.3	Complexity Classes	15
2.2.4	Approximation Algorithm	17
2.3	Probability Theory	18
2.3.1	Random Variable	18
2.3.2	Probability Density Function	18
2.3.3	Cumulative Distribution Function	19
2.3.4	Expected Value	19
2.3.5	Uniform Random Variable	19
2.3.6	Histogram	19
3	r-Gatherings on a Star	20
3.1	Preliminaries	20
3.2	r -Gather Clustering on a Star	22
3.3	r -Gathering on a Star	29
3.4	Min-Tree r -Gathering Problem	35
3.5	Summary	37
4	r-Gatherings on Uncertain Data	38
4.1	Preliminaries	39
4.2	Uncertain r -Gathering on a Line	39
4.2.1	Histogram	40
4.2.2	Uniform Distribution	48
4.3	Summary	58
5	Conclusion	61

List of Publications	63
References	64
Index	69

List of Figures

1.1	Illustration of r -gathering.	4
2.1	(a) A graph and (b) a multigraph.	11
2.2	A directed graph.	12
2.3	A tree.	13
2.4	A star graph.	14
3.1	Illustration of an r -gather clustering on a star.	22
3.2	Illustration of proof of Lemma 3.2.2.	23
3.3	Illustration of proof of Lemma 3.2.3.	24
3.4	Illustration of proof of Lemma 3.3.1.	30
4.1	Illustration of histogram and function of expected distance.	40
4.2	Illustration of uniform distribution and function of expected distance.	49

Acknowledgements

First of all, I would like to declare that all the appraisals belong to the Almighty **ALLAH**.

I would like to express my deep gratitude to my supervisor Professor Dr. Md. Saidur Rahman for introducing me to the research works in the field of Theoretical Computer Science, in particular to the clustering and facility location problems. I thank him for providing encouragement, ways of thinking and suggesting corrections in initial drafts. His constant motivation is one of the pivotal reasons behind the success of the thesis.

I would like to provide my indebtedness to Professor Dr. Shin-ichi Nakano from Gunma University for providing ideas and reviewing the drafts so many times. I would also want to thank the members of my thesis committee for their valuable suggestions. I thank Professor Dr. Md. Mostofa Akbar, Professor Dr. M. Kaykobad, Dr. Atif Hasan Rahman, and specially the external member Professor Dr. Md. Rezaul Karim.

I convey my heartfelt reverence to my parents and other family members for giving their best support throughout my work to overcome the tedium of repetitive trials to new findings.

Abstract

Given a set of customers C , a set of proposed facilities F , an opening cost $op(f)$ of each facility $f \in F$ and a cost function $cost(c, f)$ for each $c \in C$ and $f \in F$, the facility location problem asks to find an assignment of customers to facilities such that a designated cost is minimized. Because of the absence of lower bound on the number of customers assigned to each facility, the number of customers assigned to a facility can be small. To address this problem, recently a new variant of facility location problem called the r -gathering problem is introduced. An r -gathering is an assignment of customers to the facilities such that each facility serves zero or at least r customers. The r -gathering problem asks to find an r -gathering which minimizes the maximum distance between a customer and its facility. A seemingly related problem called the r -gather clustering problem asks to partition a set of points P into some clusters such that each cluster has at least r points and the maximum distance between two points in the same cluster is minimum.

Both the r -gathering and r -gather clustering problems can be used to find locations for establishing shelters with limited capacity and an assignment of residents to shelters that can minimize the evacuation span in time of disaster. Both the problems are NP-hard in general, and polynomial time algorithms are known when the customers and facilities are on a line. In this thesis, we attempt to reduce the distance between the two results. We consider r -gathering and r -gather clustering problems when the customer and facilities are on a “star” and give algorithms for the problems which run in polynomial time if the degree of the star is constant. We also consider the uncertain r -gathering problem where the customer locations are given as probability density functions. We give an exact exponential algorithm for the uncertain r -gathering problem on a line when the customer locations are specified by histograms. We also give a polynomial time algorithm for a restricted case when customer locations are given as well-separated uniform distributions.

Chapter 1

Introduction

Making better decisions is one of the keys for improvement of human from both personal to collective viewpoint. A decision making process is a problem solving activity to pick a choice from some alternatives which results in the most desired outcome. *Operations Research* is a field which deals with development and application of mathematical and algorithmic methods to make better decisions efficiently. Most problems in operations research deal with difficulties arising from managing complex management and production environment. Thus the problems mostly ask for satisfying some constraints and optimizing one or more criteria. Since the problems usually consist of many constraints and variables, efficient computational means are often required to solve those problems. Thus many problems in operations research are actively studied in computer science, mostly in the form of combinatorial optimization problems.

The *Facility Location Problem* is a well known combinatorial optimization problem in operations research and many variants of the problem are well studied [15]. Given a set of customers C , a set of facilities F , an opening cost $op(f)$ for each facility $f \in F$, a connection cost $cost(c, f)$ between each facility $f \in F$ to each customer $c \in C$, the basic facility location problem asks to open a subset $F' \subseteq F$ of facilities and find an assignment $A : C \rightarrow F$ such that a designated cost is minimum. When the connection cost between each facility and customer admits symmetry and triangle inequality then the problem is called the metric facility location problem. The facility location problem models several practical scenarios where server of certain category

should be opened to serve the clients. Such applications can be found in locating warehouse, power-plant or servers of communication networks. The most common form of the facility location problem is the *uncapacitated facility location problem* which asks to minimize the total cost of setting up facilities and serving all the customers [13]. The other common facility location problem is the *minimax facility location problem* which seeks to minimize the maximum cost incurred by setting up facilities and serving the customers [16]. All these variants of the facility location problem are uncapacitated. Thus an optimal assignment may assign a large number of customers to a facility while leaving a small number of customers to another facility. The *capacitated facility location problem* put an additional constraint on the maximum capacity of each open facility. Thus the capacitated facility location problem asks to find an assignment of customers to facilities so that no facility becomes overcrowded. However, an optimal solution of the capacitated facility location problem can have an underutilized facility because of the absence of lower bounds on the number of customers at each facility. A recently proposed variant of the facility location problem called the r -gathering problem addresses the issue of unbalanced assignment of customers to facilities by giving a lower bound on the number of customers for each open facility [10]. Since the introduction of the problem, it has gained much attention from the researcher for its practical applications and load balancing ability.

Many other variants of the facility location problem such as k -center, k -median problems [30, 17] are well studied. These problems ask to find k points as facility and assign each customer to its closest facility so that a designated cost is minimized. The designated cost is mostly defined as the total distance from each customer to its nearest facility or the maximum distance between a customer to its nearest facility. These variants often do not include any facilities as part of the input. Thus they can be viewed as a clustering problem where the customers assigned to the same facility are regarded as a cluster. However, the lack of load balancing between the facility still exists in such k -center and k -median problems. A recently introduced problem called the r -gather clustering problem can address the load balancing issue by putting an additional constraint that each cluster must have at least r customers. The r -gather clustering problem was originally introduced to protect privacy of published data through

anonymity [6]. Despite being a relatively new problem, the *r*-gather clustering problem has been studied in different settings by the researchers.

In this thesis, we study the *r*-gathering problem and the *r*-gather clustering problem. Both problems are NP-complete in general and constant factor approximation algorithms are known for them. Recently they have been considered in a restricted setting where the customers and the facilities are on a line. In this thesis, we consider a more general setting where the customers and facilities are on a “star”. In the study of facility location problems, uncertainty often plays an important role. Since facility setup is costly and each facility is intended to serve for a long time, many facility location problems are considered when the input points contain uncertainty. In this thesis, we consider the *r*-gathering problem when the customer locations are uncertain.

In the rest of this chapter, we provide the necessary background and objectives for the *r*-gathering and the *r*-gather clustering problems. In Section 1.1 we describe the *r*-gathering problem and in Section 1.2 we describe the *r*-gather clustering problem. We discuss the applications of the *r*-gathering problem and the *r*-gather clustering problem in Section 1.3 and 1.4 respectively. We devote Section 1.5 for the literature review. In Section 1.6 we detail the scope of this thesis. Finally, Section 1.7 gives the summary of the contributions of this work and Section 1.8 is the description of the organization of this thesis.

1.1 *r*-Gathering Problem

Let C be a set of n customers, F be a set of m facilities, and $d(c, f)$ be the distance between $c \in C$ and $f \in F$. An *r*-gathering of C to F is an assignment $A : C \rightarrow F$ such that each facility has at least r or zero customers assigned to it. A facility is called *open* if it has r or more customers assigned to it. The cost of an open facility f is calculated as $cost(f) = \max_{c:A(c)=f} \{d(f, c)\}$. The cost of an *r*-gathering is $\max_{c \in C} \{d(c, A(c))\}$ which is the maximum distance between a customer and its facility. The *r*-gathering problem asks to find an assignment of C to F having the minimum cost [10], and such an *r*-gathering is called an optimal *r*-gathering. This problem is also known as the min-max *r*-gathering problem. Figure 1.1(b) illustrates an optimal *r*-

gathering of the set of customers and facilities shown in Figure 1.1(a), where the customers c_1, c_2, c_3, c_4 are assigned to f_1 and c_5, c_6, c_7 are assigned to f_3 . Figure 1.1(c) illustrates another r -gathering, which is not optimal, where the customers c_1, c_2, c_3, c_4 are assigned to f_2 and c_5, c_6, c_7 are assigned to f_3 .

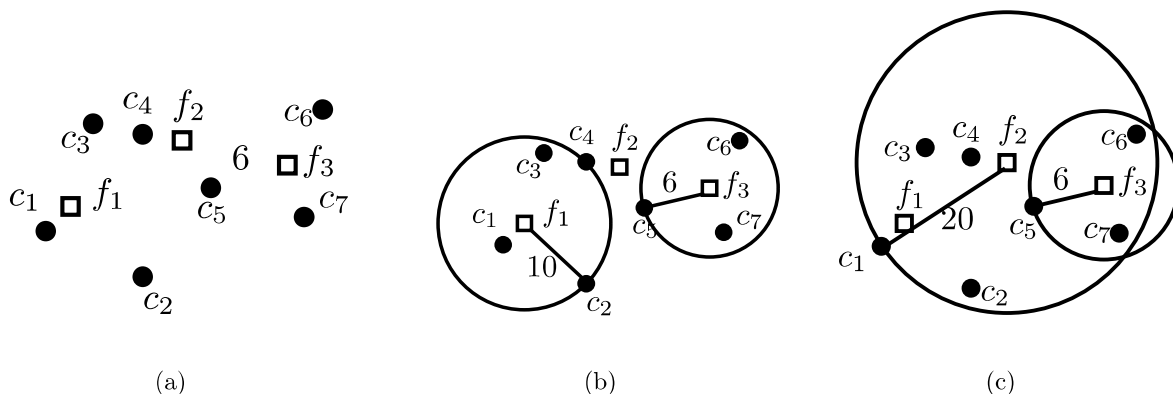


Figure 1.1: (a) A set of customers and a set of facilities, (b) an optimal r -gathering of cost 10 and (c) an r -gathering of cost 20.

Problem r -GATHERING PROBLEM

Input A set of customers C and a set of facilities F .

Output An assignment of C to F such that each facility serves zero or at least r customers and the maximum distance between a customer to its corresponding facility is minimum.

The other version of the problem is known as the min-sum r -gathering problem which asks to find an assignment which minimizes $\sum_{c \in C} d(c, A(c))$ [25, 22]. In this thesis, we consider the min-max r -gathering problem and we use the term r -gathering problem to refer the min-max version unless specified otherwise.

1.2 r -Gather Clustering Problem

Let C be a set of n points. An r -gather clustering of C is a partition of C into clusters such that each cluster contains at least r points. The cost of a cluster is the maximum distance between a pair of points in the cluster. The cost of an r -gather clustering is the maximum cost among

the costs of the clusters. The r -gather clustering problem asks to find an r -gather clustering of C with minimum cost [6], and such a clustering is called an optimal r -gather clustering.

Problem r -GATHER CLUSTERING PROBLEM

Input A set of points C .

Output A partition of C into clusters such that each cluster contains at least r points and the maximum distance between a pair of points in the same cluster is minimum.

The r -gather clustering problem can be viewed as a special case of the r -gathering problem when facilities can be placed anywhere. In such case, an optimal r -gathering can be found by placing facility in the center of the disk of minimum radius which contains all the customers of a cluster in the optimal r -gather clustering. On the other hand, for the r -gathering problem we can view the customers assigned to the same facility as a cluster of customers.

1.3 Application of r -Gathering

The r -gathering problem has application in finding suitable locations for setting up facilities such that each facility can be economically justified and the maximum distance between the customers and their corresponding facilities is minimum. Consider a scenario where kids live in a locality and there is a set of proposed playground locations in the locality. We want to set up playgrounds for the kids and assign each kid to a playground. In such situation we want each playground to be load balanced so that neither any playground becomes overpopulated nor any playground becomes underutilized. We also want to assign the kids to the playgrounds such that the time for all kids to return their homes after the game is as small as possible. In order to the playgrounds to be load balanced we give a constraint that at least r kids have to be assigned to each playground. To minimize the time for the kids to return home, we need to minimize the maximum distance between a kid and its corresponding playground. The scenario can be modeled to the r -gathering problem. An r -gathering corresponds to an assignment of kids to playgrounds such that at least r kids play in each open playground and the r -gathering

problem finds the r -gathering minimizing the time for all kids to return home.

1.4 Application of r -Gather Clustering

The r -gather clustering problem has similar application as the r -gathering problem. For example, consider the scenario described in Section 1.3. If there is no proposed playground location, then we can set up a playground in the center of the disk of minimum radius containing all kids of a cluster in an r -gather clustering and assign each kid of the cluster to the playground. Thus an r -gather clustering corresponds to an assignment of kids to playgrounds such that each “open” playground is used by at least r kids and the r -gather clustering problem finds the r -gather clustering minimizing the time for all kids to return home.

The r -gather clustering problem can also be used to ensure privacy through anonymity [6]. In particular, r -gather clustering can be used to provide location privacy in wireless networking. The ubiquity of GPS receiver in devices facilitates different location based services. For location based services, locations of devices are collected and for different queries locations are needed to be given as input to the query. In such cases, if an adversary compromises the server of a location based service, then sensitive information of users can be inferred. This yields privacy issues which can be categorized into *query privacy* and *location privacy*. The query privacy is about protecting the identity of the user requesting the query, while the location privacy is about protecting the location of the user [36, 12]. Anonymity is a common approach to ensure privacy while publishing database. One way to achieve anonymity is by *k-anonymity* where some of the identifying attributes are suppressed or generalized so that, for each record in the modified table, there are at least $k - 1$ other records in the modified table that are identical to it along the identifying attributes [41]. For query and location privacy, another approach is to group queries or locations into a cluster, each of having at least r queries or points. Thus creating clusters of queries before sending the query to the server can make the query sender indistinguishable from the other senders of the cluster. To facilitate the query processing it is desirable to create query clusters so that nearby points remain in the same

cluster. Thus the goal becomes to create clusters such that each cluster contains at least r points and maximum diameter among the disks of minimum diameter containing points of the same cluster is minimized, which can be found by an optimal r -gather clustering.

1.5 Literature Review

The facility location problem is a well studied combinatorial optimization problem [15]. The Fermat-Weber problem is considered the first facility location problem, studied as early as in the 17th century. Since then, many variants of the facility location problems have been extensively studied. Most of the variants are NP-Hard in general and many approximation algorithms are known for them [37, 21, 29, 11]. The current best approximation algorithm is due to Li who improved the approximation ratio to 1.488 [27]. On the negative side, Guha and Khuller proved that the problem cannot be approximated within a factor 1.463 unless $P = NP$ [21]. A 2-approximation algorithm is known for the capacitated facility location problem [29].

The r -gathering problem was introduced by Karger and Minkoff [25] and Guha et al. [22] in parallel [25, 22]. They both considered the min-sum r -gathering problem with different customer demands and gave a $(\frac{1+\alpha}{1-\alpha}\beta, \alpha)$ bicriteria approximation algorithm where $\alpha < 1$ and β is the approximation ratio of the metric facility location problem. A 448-approximation algorithm and 82.6-approximation algorithm for the min-sum r -gathering problem are known when each customer has unit demand [40, 7]. Recently Li gave a 4000-approximation algorithm for the facility location problem with general lower bounds [28]. The min-max r -gathering problem is NP-complete in general [10]. A 3-approximation algorithm is known for the r -gathering problem and it is proved that the problem cannot be approximated within a factor less than 3 for $r > 3$ unless $P = NP$ [10]. Recently, the r -gathering problem is considered in a setting where all the customers and facilities are lying on a line. For the r -gathering problem an $O((n+m)\log(n+m))$ time algorithm [9] based on matrix search method [18, 5], an $O(n+m\log^2 r+m\log m)$ time algorithm [23], an $O(n+r^2m)$ time algorithm [31] by reduction to the min-max path problem in a weighted directed graph [19], and an $O(n+m)$ time algorithm

[35] are known when all the customers and facilities are on a line.

The r -gather clustering problem is introduced by Aggarwal et al. [6]. They showed the hardness of the problem and gave a 2-approximation algorithm [6]. Recently, the problem is considered in a setting where the points are lying on a line. An $O(n \log n)$ time algorithm [9] based on matrix search method [18, 5], and an $O(rn)$ time algorithm [31] by reduction to the min-max path problem in a weighted directed graph [19] are known for r -gather clustering problem when all the points are on a line. Due to the linear time algorithm for the r -gathering problem [35], the r -gather clustering problem can be solved in $O(n)$ time. Recently the r -gather clustering problem is studied on mobile setting and a 4-approximation distributed algorithm is known [46]. Similar clustering problems with lower bounds on the number of points in each cluster are also studied with different cost function [1, 8].

1.6 Scope of The Thesis

The r -gathering problem and the r -gather clustering problem arise from the need of setting up economically justified, load-balanced facilities. In general settings both problems are computationally hard, while they can be efficiently solved when the input points are on a line. The algorithms for the problems when the points are on a line relies on a property that, there is an optimal solution where the points in a cluster or the customers assigned to the same facility are consecutive. However, such property does not hold when the points are on more general settings such as points on tree. In this thesis, we attempt to put the first step towards that question and consider the r -gathering problem and the r -gather clustering problem on “star”.

Since setting up a facility is costly and each facility will be justified if it can serve for a long term, it is important to consider the existence and movement of customers in the r -gathering problem. Many variants of the facility location problem are considered when the customer locations or demands contains probabilistic value. In this thesis, we consider the r -gathering problem when the locations of the customers are defined by probability density functions instead of exact locations.

1.7 Contributions

In this thesis, we consider the r -gathering problem and the r -gather clustering problem in two different settings. Firstly, we consider the problems when the input points are on a “star”. Secondly, we considered the r -gathering problem when the input points contain uncertainty and are lying on a line. The main results of this thesis are as follows.

1. We give an algorithm to solve the r -gathering problem and the r -gather clustering problem when the customers and the facilities are on a star.
2. We introduce a new cost function for the r -gathering problem and show the hardness of the problem under the new cost function even when the customers and the facilities are on a star.
3. We give an exact exponential algorithm to solve the r -gathering problem under uncertainty when the customers and the facilities are on a line and the customer locations are specified by piecewise uniform functions.
4. We give an algorithm to solve uncertain r -gathering problem when the facilities are on a line and the probability density functions of the customers are specified by “well-separated” uniform distribution functions on the line.

1.8 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, we give some basic terminology of graph theory, algorithms, complexity classes and probability theory. In Chapter 3, we present algorithms for the r -gathering and r -gather clustering problems on a star, introduce a new cost function for the r -gathering problem and show the hardness of the r -gathering problem under the new cost function. In Chapter 4, we give our results on the r -gathering problem under uncertainty. Finally, Chapter 5 discusses the open problems in this field and gives this thesis an ending.

Chapter 2

Preliminaries

In this chapter we define some basic terminologies of graph theory, algorithm, complexity classes, and probability theory. Definitions that are not included in this chapter will be introduced as they are needed. At first in Section 2.1 we define some basic terminologies of graph theory. In Section 2.2 we define some basic terminologies of algorithms and complexity classes. Finally, in Section 2.3 we define basic terminologies of probability theory.

2.1 Basic Terminology

In this section we give definitions of some graph theoretical terms used throughout the remainder of this thesis. Interested readers are referred to detailed texts of the literature [44, 32, 33].

2.1.1 Graphs

A *graph* G is a structure (V, E) which consists of a finite set of *vertices* V and a finite set of *edges* E ; each edge is an unordered pair of vertices [33]. The sets of vertices and edges of G are denoted by $V(G)$ and $E(G)$ respectively. Figure 2.1(a) depicts a graph where $V(G) = \{v_1, v_2, \dots, v_6\}$ and $E(G) = \{e_1, e_2, \dots, e_8\}$. An edge between two vertices u and v is denoted by (u, v) . If $(u, v) \in E$ then the vertices u and v are said to be *adjacent* and the edge (u, v) is said to be

incident to vertices u and v . The vertices u and v are called *end-vertices* of the edge (u, v) . The vertex u is also called a *neighbor* of v in G and vice versa. The *degree* of a vertex v in G is the number of edges incident to it in G and is denoted by $deg(v)$.

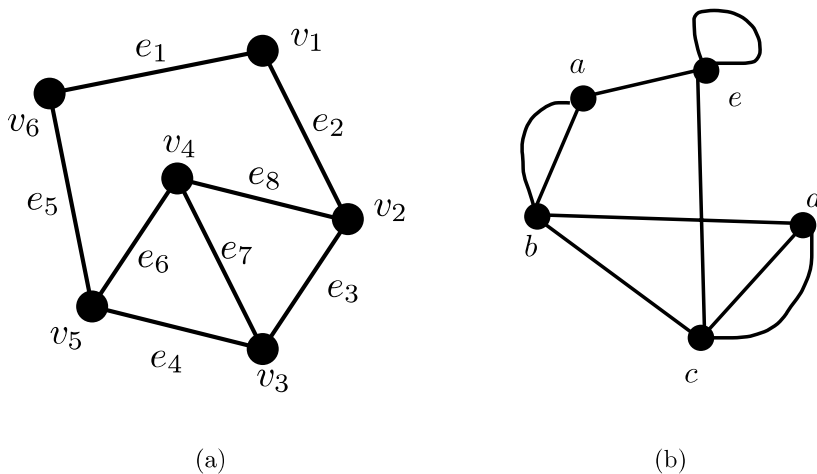


Figure 2.1: (a) A graph with six vertices and eight edges and (b) a multigraph.

A *loop* is an edge whose end-vertices are same. *Multiple edges* are edges with the same pair of end-vertices. A graph with no loop or multiple edges is called a *simple graph*. The graph shown in Figure 2.1(a) is a simple graph. On the other hand, a graph having loops or multiple edges is called a *multigraph*. Figure 2.1(b) depicts a multigraph where there is multiple edges between vertex a, b and there is a loop in vertex a . In this thesis, we only consider simple graphs.

2.1.2 Directed and Undirected Graphs

In a *directed graph*, the edges do have a direction but in an *undirected graph*, the edges do not have any direction [33]. The graph shown in Figure 2.1(a) is an undirected graph, while the graph shown in Figure 2.2 is a directed graph. In this thesis, by the term “a graph” we mean a simple, undirected graph unless otherwise mentioned.

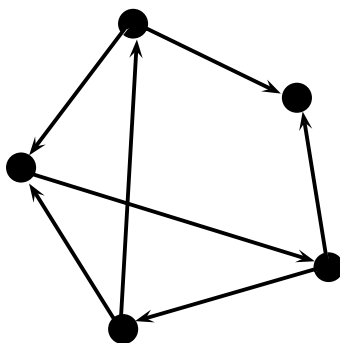


Figure 2.2: A directed graph.

2.1.3 Paths and Cycles

A *walk* in a graph G is a non-empty list $W = v_0, e_1, v_1, \dots, v_{n-1}, e_n, v_n$ whose elements are alternatively vertices and edges of G where the edge e_i has end-vertices v_{i-1} and v_i for $1 \leq i \leq n$ [33]. The vertices v_0 and v_n are called *end-vertices* of the walk W and the other vertices are called *internal vertices*. If the vertices on a walk are distinct (except end-vertices) then it is called a *path* and is usually denoted by a sequence of vertices $v_0, v_1, v_2, \dots, v_n$. If u, v are the end-vertices of a path, then it is called an u, v -path in G and we denote it as P_{uv} . A *sub-path* of P_{uv} is a subsequence $P_{v_i v_j} = v_i, v_{i+1}, \dots, v_j$ for some $1 \leq i < j \leq n$. If the end-vertices of a path or walk repeat then it is closed. A closed path containing at least one edge is called a *cycle*. A cycle with n vertices is denoted by C_n .

2.1.4 Complete Graphs

A *complete graph* is a simple graph whose vertices are pairwise adjacent. A complete graph with n vertices is denoted as K_n [33].

2.1.5 Trees

A *tree* is a connected graph that contains no cycle. Figure 2.3 depicts a tree with 13 vertices. The vertices in a tree are usually called *nodes* [33]. A *rooted tree* is a tree in which one of the

nodes is distinguished from other nodes and the node is called *root* of the tree. A rooted tree is usually drawn in a way such that the root remains in the top. In Figure 2.3, v_1 is the root. If a rooted tree is regarded as a directed graph in which each edge is directed from top to bottom, then every node u other than the root is connected by an edge from some other node p , called the *parent* of u . We also call u a *child* of node p . We draw the parent of a node above that node. In Figure 2.3, v_1 is the parent of v_2, v_3 and v_4 ; v_2, v_3 and v_4 are children of v_1 . A vertex with degree one in a tree is called a *leaf* of the tree. All the vertices other than leaves are called *internal nodes*. Thus every node of a tree is either a leaf or an internal node. In Figure 2.3, the leaves are $v_3, v_5, v_7, v_9, v_{10}, v_{11}, v_{12}$ and v_{13} , and the internal nodes are v_1, v_2, v_4, v_6 and v_8 .

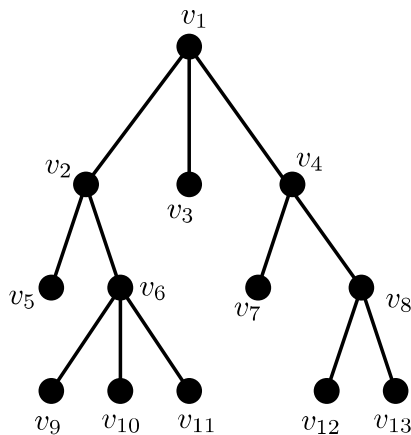


Figure 2.3: A tree.

2.1.6 Star graphs

A star graph S_n is a tree on n nodes with one node having degree $n - 1$ and all other nodes having degree 1 [33]. Figure 2.4 shows a star graph with six vertices.

2.2 Algorithms and Complexity

In this section we briefly introduce some terminologies related to algorithms and complexity of algorithms. For interested readers, we refer to [20, 14, 26].

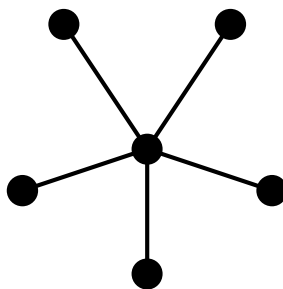


Figure 2.4: A star graph.

2.2.1 Big- O Notation

Analyzing the efficiency of an algorithm is an integral part of designing an algorithm. The standard analysis of algorithms consists of measuring *time complexity* and *memory complexity*. *Time complexity* or *running time* is the number of operations it performs before producing the output. Expressing running time in terms of basic computer steps is a simplification, since the time taken by one such step depends on the machine executing the algorithm. Thus the time for execution of an algorithm can differ from one machine to another. Even such time may differ in the same machine on another execution because of context switching, caching etc. It therefore makes more sense to seek a machine-independent characterization of the efficiency of an algorithm. To this end, we will always express running time by counting the number of basic computer steps, as a function of the size of the input. However, the time complexity of an algorithm can be made simpler. Instead of reporting that an algorithm takes, we can only consider the “asymptotic behaviour” of an algorithm. For example, let an algorithm takes $3n^2 + 5n + 2$ steps on an input of size n . Firstly, it is much simpler to leave out lower-order terms such as $5n$ and 2 since they become insignificant as n grows. Secondly, we can ignore the detail of the coefficient 3 of n^2 and just say that the algorithm takes time $O(n^2)$ (pronounced “big oh of n^2 ”). Such asymptotic behavior is more important, since when applied to very large inputs the constant factors and low order terms become insignificant. Formally, if $f(n)$ and $g(n)$ are the functions from positive integers to positive reals, then we write $f(n) = O(g(n))$ (which means that $f(n)$ grows no faster than $g(n)$) if there exists positive constants c_1 and c_2

such that $f(n) \leq c_1g(n) + c_2$ for all n .

2.2.2 Polynomial Algorithms

An algorithm is said to be *polynomially bounded* (or simply *polynomial*) if its complexity is bounded by a polynomial of the size of input [26]. Examples of such complexities are $O(n)$, $O(n \log n)$, $O(n^{100})$, etc. The remaining algorithms are usually referred to as *exponential* or *nonpolynomial*. Examples of such complexity are $O(2^n)$, $O(n!)$, etc. If the time complexity of an algorithm is bounded by $O(n)$, then we call it a *linear-time* algorithm or simply a *linear* algorithm. By an efficient algorithm, we generally mean that the algorithm is polynomial.

2.2.3 Complexity Classes

Although it is desirable to have efficient algorithm for every computational problem, there are some interesting problems for which we do not know any polynomial time algorithm yet. In this section we will briefly explain the properties of those seemingly “hard” problems and the classes of problems. In the following, we first define some terminology and the complexity classes.

Decision problems refer to the algorithmic questions that can be answered by yes or no [26]. For example, “Is there a truth assignment that satisfies a given boolean formula?” The state of algorithms consists of the values of all the variables and position of the next instruction to be executed. A *deterministic algorithm* is one for which each state, upon execution of the instruction, uniquely determines at most one of the following states (next states). The computers we have now are deterministic. On the other hand a *nondeterministic algorithm* is one for which a state may determine many next states simultaneously. A nondeterministic algorithm can be viewed as having the capability of branching off into many copies of itself, one for the each next state. Thus, while a deterministic algorithm must explore a set of alternatives one at a time, a nondeterministic algorithm examines all alternatives at the same time.

A problem Q is *polynomial time verifiable* if there is a polynomial deterministic algorithm A that takes as input the given instance \mathcal{I} of the problem Q and the proposed solution S , and

outputs true if and only if S really is a solution to instance \mathcal{I} . A problem Q is *polynomial time reducible* to problem Q' ($Q \leq_p Q'$) if there is a polynomial time algorithm that transforms an arbitrary instance \mathcal{I} of Q to an instance \mathcal{I}' of Q' such that the answer to \mathcal{I} is affirmative ($\mathcal{I} \in Q$) if and only if the answer to \mathcal{I}' is affirmative ($\mathcal{I}' \in Q'$).

The Class P

P is the class of problems that can be solved by deterministic polynomial time algorithm [26]. Thus there is a deterministic algorithm that takes as input an instance \mathcal{I} and runs in polynomial time such that if I has a solution, the algorithm returns such a solution; and if I has no solution, the algorithm correctly reports so.

The Class NP

NP stands for “nondeterministic polynomial time” [26]. The class NP contains the problems solutions of which can be verified deterministically in polynomial time. We can also define NP as the class of decision problems that can be solved nondeterministically in polynomial time. It is obvious that, $P \subseteq NP$. But the question, “ $P = NP?$ ” is still unresolved. It is widely believed that $P \neq NP$. However, proving this has turned out to be extremely difficult, one of the deepest and most important unsolved puzzles of mathematics.

The class NP-complete

A problem Q is NP-complete if it satisfies the following two conditions [26].

1. $Q \in \text{NP}$.
2. For every problem $Q' \in \text{NP}$, $Q' \leq_p Q$.

A problem satisfying condition 2 is said to be *NP-hard*, whether or not it satisfies condition 1. NP-complete problems are considered to be the hardest problems in NP. These problems have the following interesting properties.

- (a) No NP-Complete problem can be solved by any known polynomial algorithm.
- (b) If there is a polynomial algorithm for any NP-Complete problem, then all NP-complete problems can be solved in polynomial time.

2.2.4 Approximation Algorithm

Many problems with real life applications are NP-complete. Thus it is more likely that those problems are not solvable in polynomial time. However, because of practical important it is desirable to solve those problems computationally. Different approaches are known to cope with such hardness. One is developing approximation algorithm. Briefly *approximation algorithms* are polynomial algorithms for the hard problems which cannot ensure optimality of solution, rather they produce near optimal solution. This near optimal solution are often good enough. Performance of an approximation algorithm is often measured by how close the output of the approximation algorithm to the optimal solution in worst case.

Optimization problems can be of two types: maximization problems and minimization problems. For maximization problem any solution which is not optimal exhibits a smaller value than the optimum, while for minimization problem any solution which is not optimal exhibits larger cost than the optimum. We define *approximation ratio* $\rho(n)$ of an algorithm as follows: for any input of size n , the cost C of the solution produced by the algorithm is within a factor of $\rho(n)$ of the cost C^* of an optimal solution. Mathematically,

$$\max \left\{ \frac{C}{C^*}, \frac{C^*}{C} \right\} \leq \rho(n)$$

It is also easy to observe that $\rho(n) \geq 1$. A constant factor approximation algorithm is an algorithm which produces output no worse than a constant factor of the optimal result.

2.3 Probability Theory

In this section, we briefly describe some basic concepts of probability theory. For interested readers, we refer to [34].

2.3.1 Random Variable

A random variable is a variable whose possible value is determined by a random phenomenon [34]. A random variable is usually denoted by a capital letter such as X or Y . A random variable can be of two types: discrete random variable and continuous random variable. A *discrete random variable* can take a finite or countable number of distinct values. For example, the random variable denoting the outcome of a coin flip is a discrete random variable. In contrast, a *continuous random variable* can take infinite number of values. Service time in a server is an example of a continuous random variable.

2.3.2 Probability Density Function

A continuous random variable is associated with a *probability density function*(PDF) which refer the relative probability of each value to come up [34]. The PDF associated with a random variable X is denoted by $f_X(x)$. For each $x \in R$, $f_X(x) \geq 0$. The value of PDF at a certain x does not mean the probability of the occurrence of x . The probability of a continuous random variable is defined over an interval. The probability of a continuous random variable X to be in $[a, b]$ is $Pr\{X \in [a, b]\} = \int_a^b f_X(x)dx$. An important property of PDF is the area under the curve of a PDF is 1. Thus $\int_{-\infty}^{\infty} f_X(x)dx = 1$.

Probability mass function is the discrete counterpart of probability density function. However unlike probability density function, the value of probability mass function at x , denoted by $p_X(x)$, represents the probability of occurrence of x . Like probability density function, sum of the values of probability mass function is 1.

2.3.3 Cumulative Distribution Function

Cumulative Distribution Function(CDF) of a random variable X at x , denoted by $F_X(x)$, is the probability that X will take a value less than or equal to x [34]. If X is a continuous random variable, then $F_X(x) = Pr\{X \leq x\} = \int_{-\infty}^x f_X(x)dx$. For discrete case, $F_X(x) = Pr\{X \leq x\} = \sum_{x_i \leq x} p_X(x_i)$. It is easy to observe that, $Pr\{a < X \leq b\} = F_X(b) - F_X(a)$.

2.3.4 Expected Value

Expected value of a random variable is the average or mean value of the variable [34]. Expected value of X is denoted by $E[X]$. If X is a continuous random variable then $E[X]$ is calculated as $\int_{-\infty}^{\infty} xf_X(x)dx$. For discrete case, $E[X] = \sum xP_X(x)$. Expected value of a function h of a continuous random variable X is $E[h(X)] = \int_{-\infty}^{\infty} h(x)f_X(x)dx$.

2.3.5 Uniform Random Variable

A random variable X is called *uniform random variable* over (a, b) if the corresponding probability density function is given by-

$$f_X(x) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & \text{otherwise} \end{cases}$$

2.3.6 Histogram

The probability density function of a random variable can also be a *histogram*. A histogram is a piecewise uniform function. If the area under the histogram is 1, then the histogram can be considered as a probability density function. Histogram can be used as a PDF when the PDF of the data is unknown.

Chapter 3

r -Gatherings on a Star

In this chapter, we consider the r -gathering problem and the r -gather clustering problem when the customers and the facilities are on a star. In Section 3.1 we define the two problems and relevant terminologies. In Section 3.2 we give an algorithm for r -gather clustering on a star. In Section 3.3 we give an algorithm for r -gathering on a star. In Section 3.4, we introduce a new cost function for r -gathering problem and prove the hardness of the r -gathering problem under the new cost function. Finally we give a summary in Section 3.5.

3.1 Preliminaries

In this section we define the r -gathering problem and the r -gather clustering problem on star and give some basic definitions.

Let $\mathcal{L} = \{l_1, l_2, \dots, l_d\}$ be a set of d rays where all the rays of \mathcal{L} share a common source point o . We call the set of rays \mathcal{L} a *star* and the common source point o the *center* of the star. The *degree* of a star is the number d of rays which form the star. The Euclidean distance between two points p, q is denoted by $d_E(p, q)$. We denote by $d(p, q)$ the distance between two points p, q which is measured along the rays. If p and q are both located on the same ray, then $d(p, q) = d_E(p, q)$. On the other hand, if p and q are located on different rays, then $d(p, q) = d_E(p, o) + d_E(o, q)$. A cluster consists of points from two or more rays is a *multi-ray*

cluster, otherwise a *single-ray cluster*. Two points p and q are the *end-points* of a cluster \mathcal{C} if $d(p, q) = \text{cost}(\mathcal{C})$.

Let $C = \{c_1, c_2, \dots, c_n\}$ be n points located on a star. An *r-gather clustering* of C is a partition of C into clusters such that each cluster contains at least r points. The cost of a cluster \mathcal{C} , denoted by $\text{cost}(\mathcal{C})$, is $\max_{p, q \in \mathcal{C}} d(p, q)$. The *r-gather clustering problem* asks to find an *r-gather clustering* such that the maximum cost among the costs of clusters is minimized, and such a clustering is called an optimal *r-gather clustering*.

Let $C = \{c_1, c_2, \dots, c_n\}$ be n customers and $F = \{f_1, f_2, \dots, f_m\}$ be m possible locations for facilities located on a star. An *r-gathering* of C to F is an assignment A of C to F such that each facility has zero or at least r customers. A facility having one or more customers is called an *open facility*. We denote by F' the set of open facilities. $A(c)$ denotes the facility to which a customer c is assigned in an assignment A . The cost of a facility f , denoted by $\text{cost}(f)$, is $\max\{d(f, c_i) | A(c_i) = f\}$ if f has one or more customers, and is 0 if f has no customer. The *r-gathering problem* asks to find an *r-gathering* such that the maximum cost among the costs of facilities is minimized.

Consider a scenario where a number of streets meet in a junction, and residents live by the streets. We wish to set up emergency shelters on the streets so that each shelter can serve at least r residents. The distance between two points are measured along the streets. We also wish to locate shelters so that evacuation time span can be minimized. This scenario can be modeled by the *r-gather clustering problem* where all input points C are located on a star. In an *r-gather clustering* of C having the minimum cost, each emergency shelter is located at the center of each cluster. On the other hand, if the set F of possible locations of shelters on the star is also given with the set C of residents and we wish to find an assignment of C to F with minimizing the evacuation time so that each shelter serves at least r residents, then the scenario can be modeled by the *r-gathering problem* where C and F are located on a star. In this case, an *r-gathering* corresponds to an assignment of residents to shelters such that each open shelter serves at least r residents and the *r-gathering problem* finds the *r-gathering* minimizing the evacuation time.

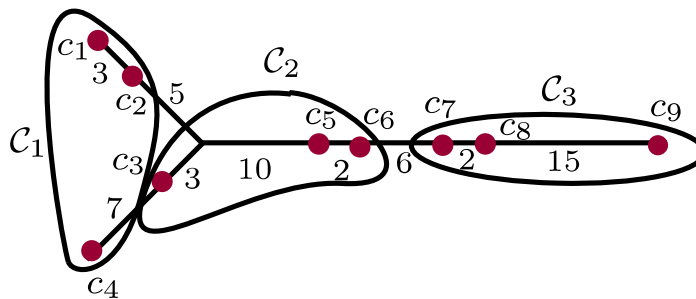


Figure 3.1: An optimal 3-gather clustering on a star.

When the points are on a line, each cluster of an optimal r -gather clustering consists of consecutive points on the line [31]. However, when the points are on a star, some clusters may not consist of consecutive points in the optimal r -gather clustering. For example, see Fig. 3.1. We can observe that at least one cluster consists of non-consecutive points in any optimal r -gather clustering. Fig. 3.1 demonstrates an optimal r -gather clustering for this scenario.

3.2 r -Gather Clustering on a Star

In this section we give an algorithm for r -gather clustering problem on a star. Let C be a set of points on a star $\mathcal{L} = \{l_1, l_2, \dots, l_d\}$ of d rays with center o . We consider the set C as a union of d sets C_1, C_2, \dots, C_d where C_i is the set of points on ray l_i . For the r -gather clustering problem, the following result is known [31]. Note that any cluster with $2r$ or more points can be divided into clusters so that each of which has at most $2r - 1$ points and at least r points.

Lemma 3.2.1 ([31]) *There is an optimal r -gather clustering in which each cluster has at most $2r - 1$ points.*

We now have the following lemma.

Lemma 3.2.2 *There is an optimal r -gather clustering such that, for each C_i , the set of points in C_i assigned to the multi-ray clusters is consecutive points on l_i including the nearest point to o .*

Proof. A pair c_m, c_s in C_i is called a reverse pair if c_m is assigned to a multi-ray cluster, c_s is assigned to a single-ray cluster, and $d(o, c_s) < d(o, c_m)$. Assume for a contradiction that A is an optimal r -gather clustering with the minimum number of reverse pairs but the number is not zero. Let c_s and c_m be a reverse pair in C_i with maximum $d(o, c_m)$. Let \mathcal{C}_s and \mathcal{C}_m be the clusters containing c_s and c_m , respectively. We have two cases.

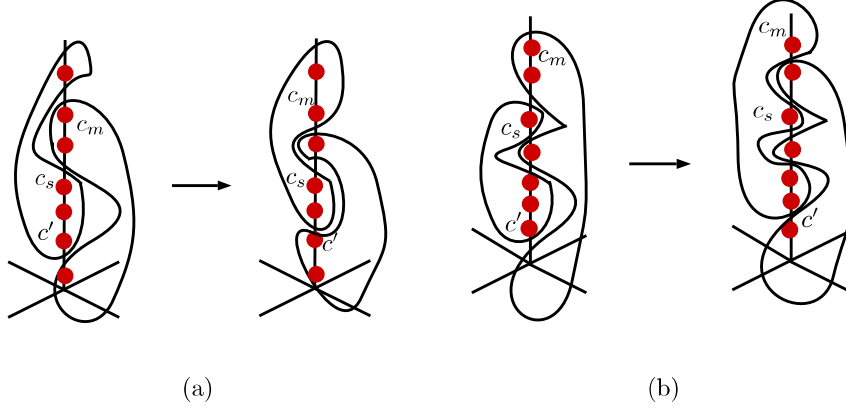


Figure 3.2: (a) Illustration of Case 1 and (b) illustration of Case 2 of proof of Lemma 3.2.2.

Case 1: \mathcal{C}_s has a point c in C_i with $d(o, c_m) < d(o, c)$.

Let c' be the nearest point to o in \mathcal{C}_s . Replacing \mathcal{C}_s and \mathcal{C}_m in the clustering by $\mathcal{C}_s \setminus \{c'\} \cup \{c_m\}$ and $\mathcal{C}_m \setminus \{c_m\} \cup \{c'\}$ generates a new r -gather clustering with less reverse pairs as illustrated in Fig. 3.2(a). A contradiction. Note that $\text{cost}(\mathcal{C}_s \setminus \{c'\} \cup \{c_m\}) \leq \text{cost}(\mathcal{C}_s)$ and $\text{cost}(\mathcal{C}_m \setminus \{c_m\} \cup \{c'\}) \leq \text{cost}(\mathcal{C}_m)$ hold.

Case 2: Otherwise. (Thus $d(o, c) < d(o, c_m)$ for every point c in \mathcal{C}_s .)

The same replacing results in a new r -gather clustering with less reverse pairs as illustrated in Fig. 3.2(b). A contradiction. Note that $\text{cost}(\mathcal{C}_s \setminus \{c'\} \cup \{c_m\}) \leq \text{cost}(\mathcal{C}_m)$ and $\text{cost}(\mathcal{C}_m \setminus \{c_m\} \cup \{c'\}) \leq \text{cost}(\mathcal{C}_m)$ hold. *Q.E.D.*

Lemma 3.2.3 *If an optimal r -gather clustering has multi-ray clusters, then at most one multi-ray cluster contains more than r points.*

Proof. Assume for a contradiction that every optimal r -gather clustering has two or more multi-ray clusters having more than r points. Let A be an r -gather clustering with the minimum

number of multi-ray clusters having more than r points. Let \mathcal{C}_i and \mathcal{C}_j be two multi-ray clusters having more than r points. Let s_i, t_i be the two endpoints of \mathcal{C}_i and s_j, t_j be the two endpoints of \mathcal{C}_j . Without loss of generality, assume that t_j is the closest point to o among the four endpoints. Let $\mathcal{C}'_j \subset \mathcal{C}_j$ be $\{c \in \mathcal{C}_j \mid d(o, c) > d(o, t_j)\}$. Any point $c \in \mathcal{C}'_j$ must be on the same ray as s_j , otherwise t_j would not be an end-point of \mathcal{C}_j . We have two cases.

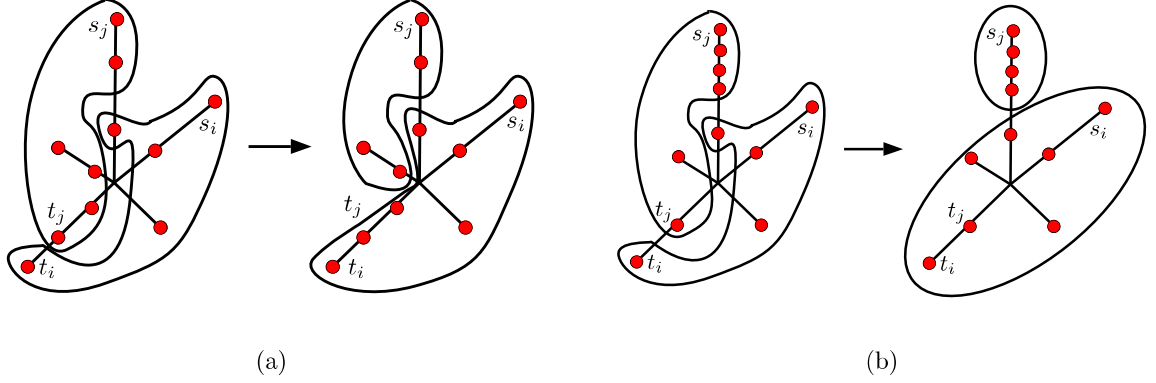


Figure 3.3: (a) Illustration of Case 1 and (b) illustration of Case 2 of proof of Lemma 3.2.3 where $r = 4$.

Case 1: $|\mathcal{C}'_j| < r$.

Let \mathcal{C}''_j be a set of $|\mathcal{C}_j| - r$ arbitrary points from $\mathcal{C}_j \setminus \mathcal{C}'_j$. We now derive a new r -gather clustering A' by replacing \mathcal{C}_i and \mathcal{C}_j by $\mathcal{C}_i \cup \mathcal{C}''_j$ and $\mathcal{C}_j \setminus \mathcal{C}''_j$. Figure 3.3(a) illustrates the construction of the new r -gather clustering. Since t_j is the closest point to o among the four end-points s_i, t_i, s_j, t_j and $d(o, c) \leq d(o, t_j)$ for any point $c \in \mathcal{C}''_j$, we have $d(o, c) \leq d(o, s_i)$ and $d(o, c) \leq d(o, t_i)$. Thus the cost of $\mathcal{C}_i \cup \mathcal{C}''_j$ does not exceed the cost of \mathcal{C}_i . Hence the cost of A' is not greater than the cost of A . Thus A' has less multi-ray clusters with more than r points, a contradiction.

Case 2: Otherwise. Thus $|\mathcal{C}'_j| \geq r$.

In this case we derive a new r -gather clustering A' by replacing \mathcal{C}_i and \mathcal{C}_j by $\mathcal{C}_i \cup (\mathcal{C}_j \setminus \mathcal{C}'_j)$ and \mathcal{C}'_j . Figure 3.3(b) illustrates the construction of the new r -gather clustering. In this case, \mathcal{C}'_j is a single-ray cluster. By a similar argument of Case 1, the cost of A' does not exceed the cost of A . Thus A' has less multi-ray clusters having more than r points than A , a contradiction. $\mathcal{Q.E.D.}$

We now give the following lemma, which is used in the proof of Lemma 3.2.5 and Lemma

3.2.7.

Lemma 3.2.4 *If $|C| \geq 2r$ and there is an optimal r -gather clustering consisting of only multi-ray clusters, then there is an optimal r -gather clustering with the multi-ray cluster consisting of the farthest point from o and its $r - 1$ nearest points.*

Proof. Let p be the farthest point from o and let N be the $r - 1$ nearest points of p . Assume for a contradiction that in every optimal solution $N \cup \{p\}$ is not a cluster. We first prove that $N \cup \{p\}$ is contained in the same cluster. Let A be an optimal solution with cluster \mathcal{C}_p containing p having the maximum number of points in N . Let q be a point in N assigned to a cluster $\mathcal{C}_q \neq \mathcal{C}_p$. Since the number of points in \mathcal{C}_p is at least r , there is a point $p' \in \mathcal{C}_p$ not in N . Let q' be the farthest point from o in $\mathcal{C}_q \setminus \{q\}$. We now derive a new r -gather clustering by replacing \mathcal{C}_p and \mathcal{C}_q by $\mathcal{C}_p \setminus \{p'\} \cup \{q\}$ and $\mathcal{C}_q \setminus \{q\} \cup \{p'\}$. Thus a contradiction. Note that, $\text{cost}(\mathcal{C}_p \setminus \{p'\} \cup \{q\}) \leq \text{cost}(\mathcal{C}_p)$ and $\text{cost}(\mathcal{C}_q \setminus \{q\} \cup \{p'\}) \leq \max\{\text{cost}(\mathcal{C}_p), \text{cost}(\mathcal{C}_q)\}$, since $d(o, p) \geq d(o, q')$.

We now prove that $N \cup \{p\}$ form a multi-ray cluster. Assume for a contradiction that in any optimal r -gather clustering $N \cup \{p\}$ is not a cluster. Let A' be an optimal r -gather clustering with cluster \mathcal{C}_p containing p having the minimum number of points not in N . Let p'' be the farthest point in \mathcal{C}_p not in the ray l_p containing p , and \mathcal{C}_s be a cluster in A' other than \mathcal{C}_p . Let s be the farthest point from o in \mathcal{C}_s . We now derive a new r -gather clustering by replacing \mathcal{C}_p and \mathcal{C}_s with $\mathcal{C}_p \setminus \{p''\}$ and $\mathcal{C}_s \cup \{p''\}$ without increasing cost, a contradiction. Since $d(o, s) \leq d(o, p)$, we have $d(s, p'') \leq d(p, p'')$ and thus $\text{cost}(\mathcal{C}_s \cup \{p''\}) \leq \max\{\text{cost}(\mathcal{C}_p), \text{cost}(\mathcal{C}_s)\}$. $\mathcal{Q.E.D.}$

We now have the following lemma.

Lemma 3.2.5 *If an optimal r -gather clustering consists of only multi-ray clusters, then there is an optimal r -gather clustering with at most $d - 1$ multi-ray clusters.*

Proof. We give a proof by induction on the number d of rays in the star. Clearly, the claim holds for $d = 2$, since in such case only one multi-ray cluster can exist. Assume that the claim

holds for any star with less than d rays. We now prove that the claim also holds for any star of d rays. Assume for a contradiction that every optimal solution has at least d multi-ray clusters. Let A be an optimal r -gather clustering with the minimum number of multi-ray clusters. Let p be the farthest point from o . By Lemma 3.2.4, there is an optimal r -gather clustering with the cluster \mathcal{C}_p containing p and its $r - 1$ nearest points, denoted by N . Let l_p be the ray containing p . We have two cases.

Case 1: p and N are on ray l_p .

In this case there is an optimal r -gather clustering with a single ray cluster $N \cup \{p\}$, a contradiction.

Case 2: Otherwise. There is a point q in N which is not on l_p .

By Lemma 3.2.4 there is an optimal r -gathering with $\{p\} \cup N$, and since N consists of the $r - 1$ nearest neighbors of p , all the points on l_p are contained in \mathcal{C}_p . Thus the points in $C \setminus \mathcal{C}_p$ are lying on other $d - 1$ rays except l_p . By inductive hypothesis there is an optimal r -gather clustering of $C \setminus \mathcal{C}_p$ with at most $d - 2$ multi-ray clusters. Thus the claim holds. $\mathcal{Q.E.D.}$

Corollary 3.2.6 *If an optimal r -gather clustering consists of only multi-ray clusters, then C has at most $(d - 2)r + 2r - 1 = dr - 1$ points.*

We now give an outline of our algorithm which constructs an optimal r -gathering clustering on a star. We first choose every possible at most $dr - 1$ candidate points for multi-ray clusters. We find the optimal r -gather clustering consisting of only multi-ray clusters for each candidate points, by repeatedly searching for the farthest point from o and its $r - 1$ nearest point as a multi-ray cluster of the remaining set of points, by the algorithm **Multi-rayClusters**.

We now have the following lemma.

Lemma 3.2.7 *Let $A = \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \dots, \mathcal{C}_{|A|}\}$ be the clusters computed by Algorithm **Multi-rayClusters**. If A has only multi-ray clusters, then A is an optimal r -gather clustering of C .*

Proof. The proof of this lemma is immediate from Lemma 3.2.4. $\mathcal{Q.E.D.}$

Algorithm 1: Multi-rayClusters(C, r)

Input : A set C of points on a star and an integer r
Output: An r -gather clustering with only multi-ray clusters
if $|C| < r$ *or the number of rays containing points is at most one* **then**
 | **return** \emptyset ;
endif
 $i \leftarrow 1$;
while $|C| \neq 0$ **do**
 | **if** $|C| < 2r$ **then**
 | Create new cluster $\mathcal{C}_i = C$;
 | **else**
 | $p \leftarrow$ farthest point from o in C ;
 | $\mathcal{C}_i \leftarrow \{p, p_1, p_2, \dots, p_{r-1}\}$ where p_i is the i -th nearest point of p in C ;
 | **endif**
 | **if** \mathcal{C}_i is a single-ray cluster **then**
 | **return** \emptyset ;
 | **endif**
 | $C \leftarrow C \setminus \mathcal{C}_i$;
 | $i \leftarrow i + 1$;
end
return $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \dots, \mathcal{C}_{i-1}\}$

We now give an algorithm ***r*GatherClusteringOnStar** to construct an optimal r -gather clustering of C on a star. We have the following theorem.

Theorem 3.2.8 *The algorithm ***r*GatherClusteringOnStar** constructs an optimal r -gather clustering of C on star in $O(n + (r + 1)^d dr)$ time.*

Proof. We first prove the correctness of the algorithm. By Lemma 3.2.2 multi-ray clusters in an optimal r -gathering are located near o , and by Corollary 3.2.6 the number of customers in the multi-ray clusters is at most $dr - 1$. The algorithm ***r*GatherClusteringOnStar** considers every possible choice of the set of points for multi-ray clusters having at most $dr - 1$ points. The algorithm considers the solution for each possible choice for multi-ray clusters with the solution obtained by algorithm for the one-dimensional setting for the remaining points on each ray, and choose the solution having minimum cost. Thus the algorithm produces an optimal r -gather clustering.

Algorithm 2: r GatherClusteringOnStar(C, r)

Input : A set C of points on star $\mathcal{L} = \{l_1, l_2, l_3, \dots, l_d\}$ and an integer r

Output: An optimal r -gather clustering of C

```

if  $|C| < r$  then
  | return  $\emptyset$ ;
endif
 $Best \leftarrow \emptyset$ ;
Let  $n_1, n_2, \dots, n_d$  be the number of points of  $C$  in each ray  $l_1, l_2, \dots, l_d$ ;
for  $i_1 \leftarrow 0$  to  $n_1$  do
  | for  $i_2 \leftarrow 0$  to  $n_2$  do
    | | for  $i_3 \leftarrow 0$  to  $n_3$  do
      | | |  $\dots$ ;
      | | | for  $i_d \leftarrow 0$  to  $n_d$  do
        | | | | if  $i_1 + i_2 + \dots + i_d < dr$  then
          | | | | |  $S$  be the set of points consisting of  $i_1, i_2, \dots, i_d$  closest points from  $o$ 
          | | | | | for ray  $l_1, l_2, \dots, l_d$ ;
          | | | | |  $R_m \leftarrow$  Multi-rayClusters( $S, r$ );
          | | | | |  $R_i \leftarrow$   $r$ -gather clustering of remaining points of ray  $l_i$  by 1D
          | | | | | algorithm;
          | | | | |  $R \leftarrow R_m \cup R_1 \cup R_2 \cup \dots \cup R_d$ ;
          | | | | | if  $R$  is the best  $r$ -gather clustering so far then
            | | | | | |  $Best \leftarrow R$ ;
          | | | | | endif
        | | | | endif
      | | | endif
    | |  $\dots$ ;
  | | end
  | | if  $i_1 + i_2 \geq dr$  then
    | | | break;
  | | endif
  | end
  | if  $i_1 \geq dr$  then
    | | break;
  | endif
end
return  $Best$ ;

```

We now estimate the running time of the algorithm. We consider points in each ray are in sorted order according to the distance from o . The d nested loops iterates $\prod_{j=1}^d (n_j + 1)$ times.

Thus the number of points involved in all calls to Multi-rayClusters is at most $(r + 1)^d dr$, since $\sum_{j=1}^d n_j = dr - 1$. Within each nested loop we repeatedly compute multi-ray clusters which takes linear time in total. We also compute single-ray clusters on each of the d rays. Rather than computing those single-ray clusters each time in the loop, we compute the r -gather clustering for points consisting of i farthest points from o , for each i , and for each ray in $O(n)$ time total [35]. Thus to compute all the required cases for single-ray cluster we need total $O(n)$ time. Thus the time complexity of the algorithm is $O(n + (r + 1)^d dr)$. *Q.E.D.*

If d is constant then this is polynomial.

3.3 r -Gathering on a Star

In this section we give an algorithm for the r -gathering problem on a star.

Let C be a set of customers and F be a set of facilities on a star $\mathcal{L} = \{l_1, l_2, \dots, l_d\}$ of d rays with center o . We regard the set C as the union of d sets C_1, C_2, \dots, C_d where C_i is the set of customers on ray l_i . Similarly, F is the union of F_1, F_2, \dots, F_d where F_i is the set of facilities on ray l_i . In any optimal r -gathering each open facility serves at least r customers. However the number of customers assigned to an open facility can be more than $2r - 1$. In such case we regard the set of customers assigned to a facility as the union of clusters $\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_k$ sharing a facility and each of which satisfies $r \leq |\mathcal{C}_i| < 2r$. Thus we can think of the r -gathering problem in a similar way to the r -gather clustering problem in Section 3, and Lemma 3.2.1 holds for the clusters of r -gathering. We denote by $A(\mathcal{C})$ the facility to which the customers in \mathcal{C} is assigned in r -gathering A . We define the cost of a cluster \mathcal{C} , denoted by $cost(\mathcal{C})$, in r -gathering A as $\max_{c \in \mathcal{C}} \{d(c, A(c))\}$. It is easy to observe that Lemma 3.2.2 also holds for the clusters of r -gatherings. We now prove that Lemma 3.2.3 also holds for the r -gathering problem.

Lemma 3.3.1 *There is an optimal r -gathering including at most one multi-ray cluster having more than r customers.*

Proof. Assume for a contradiction that every optimal r -gathering has two or more multi-ray

clusters having more than r customers. Let A be an r -gathering with the minimum number of multi-ray clusters having more than r customers. Let \mathcal{C}_i and \mathcal{C}_j be two multi-ray clusters having more than r customers. Let $f_i = A(\mathcal{C}_i)$ and $f_j = A(\mathcal{C}_j)$. Let f_i and f_j be located on ray l_i and l_j , respectively. Without loss of generality, assume that $d(o, f_i) \leq d(o, f_j)$. Let \mathcal{C}'_j be the subset of \mathcal{C}_j located on l_j . We have two cases.

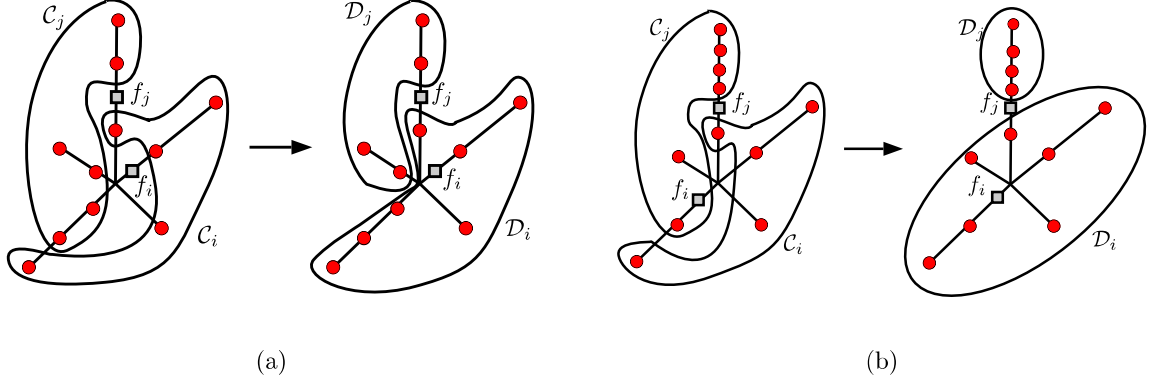


Figure 3.4: (a) Illustration of Case 1 and (b) illustration of Case 2 of proof of Lemma 3.3.1 where $r = 4$.

Case 1: $|\mathcal{C}'_j| < r$. Let \mathcal{C}''_j be a set of $|\mathcal{C}_j| - r$ arbitrary points from $\mathcal{C}_j \setminus \mathcal{C}'_j$. We now derive a new r -gathering A' by replacing \mathcal{C}_i and \mathcal{C}_j by $\mathcal{D}_i = \mathcal{C}_i \cup \mathcal{C}''_j$ and $\mathcal{D}_j = \mathcal{C}_j \setminus \mathcal{C}''_j$. Figure 3.4(a) illustrates the transformation to the new r -gathering. Note that $\mathcal{C}_j \setminus \mathcal{C}''_j$ has exactly r customers. Let c be a customer in \mathcal{C}''_j . Since f_i is closer to o than f_j and c is not on l_j , we have $d(f_i, c) \leq d(o, f_i) + d(o, c) \leq d(o, f_j) + d(o, c) = d(c, f_j)$. Thus the cost of $\mathcal{C}_i \cup \mathcal{C}''_j$ does not exceed the cost of $\max\{\text{cost}(\mathcal{C}_i), \text{cost}(\mathcal{C}_j)\}$. Hence the cost of A' is not greater than the cost of A . Thus A' has less multi-ray clusters with more than r points, a contradiction.

Case 2: Otherwise. Thus $|\mathcal{C}'_j| \geq r$. In this case we derive a new r -gathering A' by replacing \mathcal{C}_i and \mathcal{C}_j by $\mathcal{D}_i = \mathcal{C}_i \cup (\mathcal{C}_j \setminus \mathcal{C}'_j)$ and $\mathcal{D}_j = \mathcal{C}'_j$. Figure 3.4(b) illustrates the new r -gathering. In this case, \mathcal{C}'_j is a single-ray cluster. By a similar argument of Case 1, the cost of A' does not exceed the cost of A . Thus A' has less multi-ray clusters having more than r customers than A , a contradiction. *Q.E.D.*

We now give the following lemma.

Lemma 3.3.2 *If $|C| \geq 2r$ and there is an optimal r -gathering A with only multi-ray clusters, then there is an optimal r -gathering with only multi-ray clusters satisfying the following (a) and (b). Let f be the farthest open facility from o in A and l be the ray containing f .*

(a) *The farthest customer p from o on l and its $r - 1$ nearest customers form a multi-ray cluster, if l has a customer,*

(b) *All customers are assigned to f and the farthest customer p from o and its $r - 1$ nearest customers form a multi-ray cluster, if l has no customer.*

Proof. (a) We denote by N the set of the $r - 1$ nearest customers of p . We first prove that there is an optimal solution with the customers in $N \cup \{p\}$ are assigned to f . Assume for a contradiction that in any optimal solution $N \cup \{p\}$ are not assigned to f . Let A be an optimal solution with the maximum number of customers in $N \cup \{p\}$ are assigned to f . Let \mathcal{C}_p be the multi-ray cluster assigned to f , and q be a customer in $N \cup \{p\}$ but $q \notin \mathcal{C}_p$. Let q is assigned to f' . Since \mathcal{C}_p has at least r customers, there is a customer $p' \in \mathcal{C}_p$ not in $N \cup \{p\}$ and lying on a ray except l . We now derive a new r -gathering A' by reassigning q to f and p' to f' . Since $d(o, f') \leq d(o, f)$, we have $d(f', p') \leq d(o, f') + d(o, p') \leq d(o, f) + d(o, p') = d(f, p')$. Now if q is (1) not on l or (2) q is on l with $d(o, q) \leq d(o, f)$ then $d(f, q) \leq d(f, p')$. Otherwise, q is on l with $d(o, q) > d(o, f)$ holds, then we have $d(f, q) \leq d(f, p)$. Thus the cost of A' does not exceed the cost of A , and A' has more customers in $N \cup \{p\}$ assigned to f . A contradiction. Thus the customers in $N \cup \{p\}$ are contained in \mathcal{C}_p .

We now prove that $N \cup \{p\}$ form a multi-ray cluster. Assume for a contradiction that in any optimal r -gathering $N \cup \{p\}$ is not a cluster. Let A' be an optimal r -gathering with the cluster \mathcal{C}_p containing p having the minimum number of customers not in $N \cup \{p\}$. Since \mathcal{C}_p is a multi-ray cluster, \mathcal{C}_p has a customer p' not in $N \cup \{p\}$ and lying on a ray except l . Let \mathcal{C}_s be a cluster in A' other than \mathcal{C}_p and $A'(\mathcal{C}_s) = f'$. We now derive a new r -gathering by replacing \mathcal{C}_p and \mathcal{C}_s by $\mathcal{C}_p \setminus \{p''\}$ and $\mathcal{C}_s \cup \{p''\}$. Since p'' is reassigned to f' and $d(o, f') \leq d(o, f)$, $d(s, f')$ does not exceed $d(s, f)$. A contradiction.

(b) We first prove that all customers are assigned to f . Assume for a contradiction that

there is an open facility $f' \neq f$ to which some customers are assigned. Since f is the farthest open facility from o and there is no customer on l , we can reassign all customers to f' without increasing the cost of the r -gathering. A contradiction.

The proof of the 2nd part of Lemma 3.3.2(b) is similar to the proof of Lemma 3.3.2(a).

Q.E.D.

We now prove that Lemma 3.2.5 also holds for r -gathering.

Lemma 3.3.3 *If an optimal r -gathering consists of only multi-ray clusters, then there is an optimal r -gathering consisting of at most $d - 1$ multi-ray clusters, where d is the number of rays containing a customer.*

Proof. We give a proof by induction on d .

We first show the claim holds for $d = 2$. Assume for a contradiction every optimal r -gathering has two or more multi-ray clusters. Let A be an optimal r -gathering with the minimum number of multi-ray clusters, and f be the farthest open facility from o in A and l be the ray containing f . If there is no customer on l , then by Lemma 3.3.2(b) all customers are assigned to f and the farthest customer p from o in C and its $r - 1$ nearest customers N form a cluster \mathcal{C} . Otherwise by Lemma 3.3.2(a) the farthest customer p from o on l and its $r - 1$ nearest customers N form a cluster \mathcal{C} . In both case either \mathcal{C} or the other cluster is a single-ray cluster, a contradiction.

Now we consider for $d > 2$. Assume that the claim holds if the customers are on less than d rays. We now prove that the claim also holds if the customers are on exactly d rays. Assume for a contradiction that every optimal r -gathering with only multi-ray clusters has at least d multi-ray clusters. Let A be an optimal r -gathering with the minimum number of multi-ray clusters. Let f be the farthest open facility from o in A . Let l be the ray containing f . We have the following two cases.

Case 1: There is a customer on l . Let p be the farthest customer from o on l and N be the $r - 1$ nearest customers of p . By Lemma 3.3.2(a), there is an optimal r -gathering consisting of only multi-ray clusters with cluster $\mathcal{C}_p = N \cup \{p\}$. Now the customers in $C \setminus \mathcal{C}_p$ are lying

on other $d - 1$ rays except l . By inductive hypothesis there is an optimal r -gathering of $C \setminus \mathcal{C}_p$ with at most $d - 2$ multi-ray clusters. Thus the claim holds.

Case 2: Otherwise. By Lemma 3.3.2(b), there is an optimal r -gathering where all customers are assigned to f . Since there are at least d multi-ray clusters, the number of customers is at least dr . Thus there is a ray l' with r or more customers. We can form a single-ray cluster with the r customers on l , a contradiction. *Q.E.D.*

We now give algorithm Multi-rayClusters2. If there is an optimal r -gathering with only multi-ray clusters, then the algorithm finds such an r -gathering, by repeatedly removing a cluster ensured by Lemma 3.3.2.

Algorithm 3: Multi-rayClusters2(C, F, r)

Input : A set C of customers and a set of F of facilities on a star, and an integer r
Output: An r -gathering with only multi-ray clusters
if $|C| < r$ or the number of rays containing customers is at most one or $F = \emptyset$ **then**
 | **return** \emptyset ;
endif
if $|C| < 2r$ or the number of rays containing customers is two **then**
 | Assign C to its best facility; /* Lemma 3.3.2(b) */
 | **return** $\{C\}$;
endif
 $Ans \leftarrow \emptyset$;
 $Best \leftarrow \infty$;
for each ray l_i containing a customer **do**
 | $\mathcal{C}_i \leftarrow p_i$ and its $r - 1$ nearest customers in C ; /* Lemma 3.3.2(a) */
 | **if** \mathcal{C}_i is a multi-ray cluster **then**
 | Assign \mathcal{C}_i to its best facility;
 | $A \leftarrow \{\mathcal{C}_i\} \cup \text{Multi-rayClusters2}(C \setminus \mathcal{C}_i, F, r)$;
 | **if** $cost(A) < Best$ **then**
 | $Best \leftarrow cost(A)$;
 | $Ans \leftarrow A$;
 | **endif**
 | **endif**
end
return Ans

Lemma 3.3.4 *If there is an optimal r -gathering consisting of only multi-ray clusters, then Al-*

gorithm **Multi-rayClusters2** finds an optimal r -gathering. The running time of the algorithm is $O(2^d r d + (d + \log m) d^2 r^2)$.

Proof. If there is an optimal r -gathering with only multi-ray clusters, then, by repeatedly removing a cluster ensured by Lemma 3.3.2, we can find a sequence $\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_k$ of multi-ray clusters such that \mathcal{C}_i consists of exactly r customers in $C \setminus (\mathcal{C}_1 \cup \mathcal{C}_2 \cup \dots \cup \mathcal{C}_{i-1})$ except the last cluster \mathcal{C}_k with $r \leq |\mathcal{C}_k| \leq 2r - 1$. The algorithm checks every possible sequence of the rays containing the farthest open facility and chooses the best one as an optimal r -gathering. Note that if a cluster is a single-ray cluster, then the algorithm skips recursive call, since it try to find an r -gathering consisting of only multi-ray clusters.

We now estimate the running time of the algorithm.

By Lemma 3.3.3 the depth of the recursive calls is at most $d - 1$. Thus, by the tree structure of the calls, the number of calls is at most $d!$. The algorithm repeatedly constructs a multi-ray cluster with exactly r customers by Lemma 3.3.2. We can precompute the sorted list of candidate customers for the multi-ray clusters according to the distance from the center in $O(dr \log d)$ time. Using the sorted list, construction of each multi-ray cluster takes $O(r)$ time for each and $O(rd)$ time in total. The cluster is assigned to its best facility of the cluster. The best facility of a multi-ray cluster is the nearest facility to the mid-point of the farthest two customers on two different rays in the cluster. The best facility can be found in $O(d + \log m)$ time for each cluster. For each possible pair of customers we precompute the best facility. Thus the algorithm runs in $O(d! r d + (d + \log m) d^2 r^2)$ time.

We can improve the running time by modifying the algorithm to save the solution of each subproblem in a table. The number of distinct subproblems is the number of the combinations of the rays checked. Thus the number of distinct subproblems is $\sum_{j=1}^{d-1} \binom{d}{j} = O(2^d)$. Then the running time is $O(2^d r d + (d + \log m) d^2 r^2)$. *Q.E.D.*

Theorem 3.3.5 *An optimal r -gathering of C to F can be computed in $O(n + m + d^2 r^2 (d + \log m) + (r + 1)^d 2^d r d)$ time.*

Proof. Similar to Theorem 3.2.8 we can prove the number of possible choices of the customers

for multi-ray clusters is at most $(r + 1)^d dr$. For each choice we compute an r -gathering with Multi-rayClusters2 and compute r -gatherings of the remaining one-dimensional problems, then combine them to form an r -gathering of C to F . Then output the best one. This construction of multi-ray clusters needs $O(2^d rd + (d + \log m)d^2 r^2)$ for each. To eliminate redundant computation we precompute the best facilities of each pair in the dr customers which are candidate for the farthest two customers in possible multi-ray clusters. Such precomputation takes $O(d^2 r^2 (d + \log m))$ time. We can solve all possible one dimensional r -gathering problem in $O(n + m)$ time in total [35] and we store the solutions in a table. Note that when we solve one dimensional r -gathering problem of ray l , we may assign a cluster to the nearest facility to o located on other ray, however one can compute such f quickly. Thus the time complexity of finding an optimal r -gathering is $O(n + m + d^2 r^2 (d + \log m) + (r + 1)^d 2^d rd)$. Q.E.D.

If d is constant, then this is polynomial.

3.4 Min-Tree r -Gathering Problem

In this section we introduce a new cost function for the r -gathering problem and show that the r -gathering problem is NP-Hard with the new cost function even when the points are on a star.

Let C be a set of customers, F be a set of facilities and A be an r -gathering of C to F . We define the *tree cost* of a facility f as $\sum_{c:A(c)=f} d(c, A(c))$. The *min-tree r -gathering problem* asks to find an r -gathering such that the maximum tree cost among all the facilities is minimum.

The *decision min-tree r -gathering problem* is defined as follows.

Problem: DECISION MIN-TREE r -GATHERING PROBLEM.

Instance: A set of customers C and a set of facilities F , an integer r , and a number q .

Question: Does there exist an r -gathering A such that for each $f \in F$, $\sum_{c:A(c)=f} d(c, A(c)) \leq q$?

We show the hardness of the decision min-sum r -gathering problem by reduction from the *3-partition problem* [20]. The 3-partition problem is defined as follows.

Problem: 3-PARTITION PROBLEM.

Instance: A multi-set $S = \{a_1, a_2, \dots, a_{3k}\}$ of $3k$ integers and a number b such that $\frac{b}{4} < a_i < \frac{b}{2}$ for each $1 \leq i \leq 3k$ and $\sum a_i = kb$.

Question: Can S be partitioned into k subsets S_1, S_2, \dots, S_k such that for each $i = 1, 2, \dots, k$; $\sum_{a \in S_i} a = b$?

We now give the following theorem.

Theorem 3.4.1 *The decision min-tree r -gathering problem is NP-Hard even when the customers and facilities are on a star .*

Proof. We prove the NP-Hardness of the decision min-tree r -gathering problem by giving a polynomial time reduction from the 3-partition problem.

Given an instance $\mathcal{I}(S, b)$ of the 3-partition problem, we construct an instance $\mathcal{I}(C, F, r, q)$ of the decision min-tree r -gathering problem such that $\mathcal{I}(S, b)$ has an affirmative answer if and only if $\mathcal{I}(C, F, r, q)$ has an affirmative answer. We first construct a star $\mathcal{L} = \{l_1, l_2, \dots, l_{3k}\}$ of degree $3k$ and center o . For each $a_i \in S$ we take a customer c_i , lying on l_i , such that $d(o, c_i) = a_i$. Note that, $\frac{b}{4} < d(o, c_i) < \frac{b}{2}$ holds, since $\frac{b}{4} < a_i < \frac{b}{2}$ for each a_i . We now take k facilities f_1, f_2, \dots, f_k such that f_i is lying ray l_i and $d(o, f_i) = \epsilon$ where $\epsilon < \min\{d(o, c_i)\}$. Finally we set $q = b + \epsilon$ and $r = 3$. In the following we prove that there is a solution to an instance $\mathcal{I}(S, b)$ if and only if $\mathcal{I}(C, F, r, q)$ has a solution.

We first assume that $\mathcal{I}(S, b)$ has an affirmative answer. Let S_1, S_2, \dots, S_k be the partition of S such that $\sum_{a \in S_i} a = b$ for each S_i . We can construct an r -gathering of instance $\mathcal{I}(C, F, r, k)$ in the following way: for each $S_i = \{a_{i_1}, a_{i_2}, a_{i_3}\}$ we assign the customers $c_{i_1}, c_{i_2}, c_{i_3}$ to the facility f_{i_1} . Note that f_{i_1} is lying on the same ray l_{i_1} as c_{i_1} . Since there is no other customer on ray l_{i_1} , the number of customers assigned to f_{i_1} is exactly 3. Now the cost of the facility f_{i_1} is $d(o, c_{i_1}) + d(o, c_{i_2}) + d(o, c_{i_3}) + \epsilon = b + \epsilon$. Thus each open facility f_i serves at least 3 customers and for each $f \in F$, $\sum_{c: A(c)=f} d(c, A(c)) \leq b + \epsilon$.

Conversely, assume that $\mathcal{I}(C, F, r, q)$ has an affirmative answer. Let A be the corresponding r -gathering. We first claim that, each open facility in A serves exactly 3 customers. For a contradiction, assume otherwise. Let f_i be an open facility such that f_i serves at least four

customers. Since $\frac{b}{4} < d(o, c_i) < \frac{b}{2}$ holds for each c_i , $\sum_{c:A(c)=f_i} d(f_i, c) \geq \sum_{c:A(c)=f_i} d(o, c) + 2\epsilon > 4 \times \frac{b}{4} = b + 2\epsilon$, a contradiction. We now claim that, $\sum_{c:A(c)=f_i} d(f_i, c) = b$ holds for each open facility f_i . Assume for a contradiction that, $\sum_{c:A(c)=f_i} d(f_i, c) < b + \epsilon$ holds for an open facility f_i . Thus we get $\sum_{c:A(c)=f_i} d(o, c) < b$. Let C' be the set of customers assigned to some facility other than f_i . Clearly $|C'| = 3k - 3$. Since $\sum_{c \in C'} d(o, c) = kb$ and $\sum_{c:A(c)=f_i} d(f_i, c) < b$, we get $\sum_{c \in C'} d(o, c) > (k - 1)b$. Then there is at least one facility f_j for which $\sum_{c:A(c)=f_j} d(o, c) > b$ holds. Thus $\sum_{c:A(c)=f_j} d(f_j, c) > b + \epsilon$. A contradiction. *Q.E.D.*

3.5 Summary

In this chapter, we give algorithms for the r -gather clustering problem and the r -gathering problem which run in polynomial time if the number of rays are constant. We also introduce the min-tree r -gathering problem and show the hardness of the min-tree r -gathering problem even when the customers and the facilities are on a star.

Chapter 4

r -Gatherings on Uncertain Data

In this chapter we study the r -gathering problem on uncertain data.

Study of different problems under uncertain settings become much popular recently. Uncertainty in data usually occurs because of noise in measured data, sampling inaccuracy, limitation of resources etc. Hence uncertainty is ubiquitous in practice and managing the uncertain data has gained much attention [2, 3, 4, 39]. Different variants of the facility location problem has also been investigated under uncertain settings. Setting up a facility is costly to do and each facility is supposed to serve for a long period of time. On the other hand existence, location and demand of a client can change over time. Thus it is important to set up facility by keeping the uncertainty in mind. For the detailed state of art of uncertain facility location problem, we refer the survey of Snyder [38].

There are two models to deal with location uncertainty: one is existential model [24, 45] and the other is locational model [2, 3, 42]. In the existential model, existence of each point is uncertain. Thus each point has a specific location and there is a probability for the existence of each point. In the locational model each point is certain to be exist, but its position is uncertain and defined by a PDF. In this thesis we consider the locational model of uncertainty.

The rest of the chapter is organized as follows. In Section 4.1 we define the uncertain r -gathering problem and provide definitions of basic terminologies. In Section 4.2 we give algorithms for uncertain r -gathering problem for two type of input probability distributions.

Finally we provide a summary in Section 4.3.

4.1 Preliminaries

In this section we define the uncertain r -gathering problem and relevant terminologies.

Let $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of n customers where each C_i is a identically and independently distributed random variable, and $F = \{f_1, f_2, \dots, f_m\}$ be a set of m facilities. The expected distance between a facility f and an uncertain customer C , denoted by $E[d(C, f)]$, is $\int_{-\infty}^{\infty} d(x, f)g_C(x)dx$ where $g_C(x)$ is the PDF associated with C . An r -gathering A of \mathcal{C} to F is an assignment $A : \mathcal{C} \rightarrow F$ such that each facility serves zero or at least r customers. The cost of a facility is the maximum expected distance between the facility and its customers. The cost of an r -gathering is the maximum cost among all the facilities. The *uncertain r -gathering problem* asks to find an r -gathering with minimum cost. In this chapter we refer the traditional version of r -gathering problem as *deterministic r -gathering problem* in order to differentiate between the deterministic and uncertain counterpart.

It is easy to observe that the general uncertain r -gathering problem is NP-Hard since it contains the deterministic r -gathering problem as a special case.

4.2 Uncertain r -Gathering on a Line

In this section we give algorithms for the uncertain r -gathering problem on a line.

Let $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of n uncertain customer on a line where each customer is specified by a PDF $g_i : \mathbb{R} \rightarrow \mathbb{R}^+ \cup \{0\}$, and $F = \{f_1, f_2, \dots, f_m\}$ be a set of m facilities on the line. We consider the facilities are ordered from left to right. An r -gathering of \mathcal{C} to F is an assignment of $A : \mathcal{C} \rightarrow F$ such that each facility serve zero or at least r customers. The uncertain r -gathering problem asks to find an r -gathering such that maximum expected distance among the expected distances between a customer to the corresponding facility is minimum.

4.2.1 Histogram

In this section we give algorithm for uncertain r -gathering problem when each customer location is specified by a piecewise uniform function, i.e., a histogram.

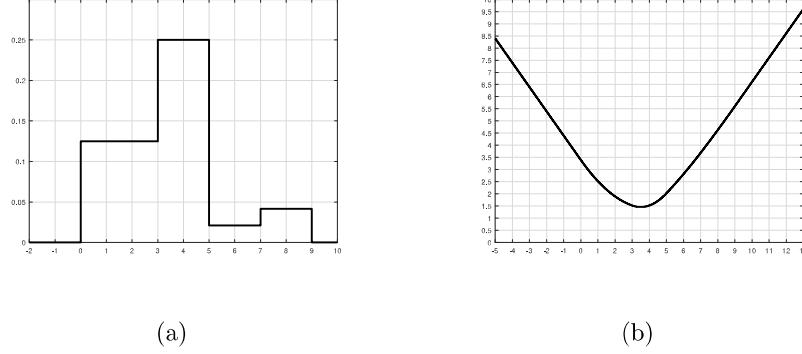


Figure 4.1: (a) Illustration of histogram as a PDF and (b) corresponding function of expected distance.

We consider the pdf g_i of each customer is defined as a piecewise uniform function, i.e., a histogram and the distribution of each uncertain customer is independent. We consider histogram since it can be used to approximate any other PDF [2]. The histogram model is considered by Wang and Zhang [43] for the uncertain k -center problem on a line. Each PDF g_i consists of at most $k+1$ pieces where each piece is a uniform function. Specifically for a customer C_i , there are k points $x_{i1}, x_{i2}, \dots, x_{ik}$ where $x_{i1} < x_{i2} < \dots < x_{ik}$, and $y_{i1}, y_{i2}, \dots, y_{i(k-1)}$ such that $g_i(x) = y_{ij}$ if $x_{ij} \leq x \leq x_{i(j+1)}$. We also consider two other points $x_{i0} = -\infty$ and $x_{i(k+1)} = \infty$, and $g_i(x) = 0$ if $x < x_{i1}$ or $x \geq x_{ik}$. Figure 4.1(a) illustrates a histogram of 6 pieces. The expected distance $E[d(p, C_i)]$ from a point p to C_i is defined as follows.

$$E[d(p, C_i)] = \int_{-\infty}^{\infty} g_i(x)|x - p|dx$$

A function $h : \mathbb{R} \rightarrow \mathbb{R}$ is called a *unimodal function* if there is a point p such that $h(x)$ is monotonically decreasing in $(-\infty, p]$ and monotonically increasing in $[p, \infty)$. Wang and Zhang showed that the function $E[d(p, C)]$ is unimodal as described in the following lemma [43].

Lemma 4.2.1 ([43]) *Let C be an uncertain point which is specified by a piece-wise uniform function consisting of $k + 1$ pieces. Then the function $E[d(p, C)]$ is a unimodal function.*

We now review the following lemma given by Wang and Zhang [43].

Lemma 4.2.2 ([43]) *Let C be an uncertain point which is specified by a piece-wise uniform function consisting of $k + 1$ pieces. Then the function $E[d(p, C)]$ consists of a parabola in each interval $[x_j, x_{j+1})$. Furthermore the function $E[d(p, C)]$ can be computed in $O(k)$ time.*

Proof. We first compute the function $E[d(p, C)]$. Without loss of generality, assume that $x_t \leq p \leq x_{t+1}$. Then we have,

$$\begin{aligned} E[d(p, C)] &= \int_{-\infty}^{\infty} |x - p|g(x)dx \\ &= \int_{-\infty}^p (p - x)g(x)dx + \int_p^{\infty} (x - p)g(x)dx \end{aligned}$$

We now evaluate the first integral. We have,

$$\begin{aligned} \int_{-\infty}^p (p - x)g(x)dx &= \sum_{j=0}^{t-1} \int_{x_j}^{x_{j+1}} (p - x)y_j dx + \int_{x_t}^p (p - x)y_t dx \\ &= \sum_{j=0}^{t-1} y_j \left(px - \frac{x^2}{2} \right)_{x_j}^{x_{j+1}} + y_t \left(px - \frac{x^2}{2} \right)_{x_t}^p \\ &= \sum_{j=0}^{t-1} y_j \left[p(x_{j+1} - x_j) - \frac{x_{j+1}^2 - x_j^2}{2} \right] + y_t \left[p(p - x_t) - \frac{p^2 - x_t^2}{2} \right] \\ &= \frac{y_t}{2}p^2 + \left[\sum_{j=0}^{t-1} y_j (x_{j+1} - x_j) - x_t y_t \right] p + \frac{x_t^2 y_t}{2} - \sum_{j=0}^{t-1} y_j \left[\frac{x_{j+1}^2 - x_j^2}{2} \right] \end{aligned}$$

Similarly for the second integral we obtain,

$$\int_{-\infty}^p (p - x)g(x)dx = \frac{y_t}{2}p^2 - \left[\sum_{j=t+1}^k y_j (x_{j+1} - x_j) - x_t y_{t+1} \right] p + \frac{x_{t+1}^2 y_t}{2} + \sum_{j=t+1}^k y_j \left[\frac{x_{j+1}^2 - x_j^2}{2} \right]$$

Thus we get,

$$\begin{aligned} E[d(p, C)] &= y_t p^2 + \left[\sum_{j=0}^{t-1} y_j (x_{j+1} - x_j) - \sum_{j=t+1}^k y_j (x_{j+1} - x_j) - y_t (x_t + x_{t+1}) \right] p \\ &\quad + \frac{1}{2} \left[\sum_{j=t+1}^k y_j (x_{j+1}^2 - x_j^2) - \sum_{j=0}^{t-1} y_j (x_{j+1}^2 - x_j^2) + y_t (x_t^2 + x_{t+1}^2) \right] \quad (4.1) \end{aligned}$$

Thus we can write $E[d(p, C)] = a_1(t)p^2 + a_2(t)p + a_3(t)$. Clearly, $E[d(p, c)]$ is a parabola in each $[x_t, x_{t+1})$. Note that if $y_t = 0$ for any $[x_t, x_{t+1})$ then the function $E[d(p, C)]$ is a line in the interval $[x_t, x_{t+1})$ which we consider as a special parabola. Figure 4.1(b) illustrates the $E[d(p, C)]$ function for the histogram in Figure 4.1(a).

We now show that the function $E[d(p, C)]$ can be computed in $O(k)$ time. Since $a_1(t) = y_t$, we can compute $a_1(t)$ of each piece in constant time. Both $a_2(t)$ and $a_3(t)$ contain terms which are either sum of terms from 0 to $t - 1$, or from $t + 1$ to k . However, we can compute those sum terms using the value computed for $a_2(t - 1)$ and $a_3(t - 1)$. Hence computation of each $a_2(t)$ and $a_3(t)$ take additional constant time. *Q.E.D.*

We now give the following lemma.

Lemma 4.2.3 *Let C be an uncertain point on a line which is specified by a piece-wise uniform function consisting of $k + 1$ pieces, and $F = \{f_1, f_2, \dots, f_m\}$ be a set of m facilities on the line. We can compute expected distances between all the facilities and the uncertain point in $O(m + k)$ time. Furthermore the expected distances between the facilities and the uncertain point can be sorted in $O(m)$ time.*

Proof. We first precompute the function $E[d(p, C)]$ in $O(k)$ time by Lemma 4.2.2. With the precomputed function $E[d(p, C)]$, the expected distance between the uncertain point and all the facilities can be computed in $O(m \log(k))$ time using binary search. However, we can improve the running time to $O(m + k)$ using a plane sweep from left to right. We take each facility in left to right order, determine the corresponding interval $[x_j, x_{j+1})$, and compute the value of $E[d(f_i, C)]$. Since both the facilities and the x_1, x_2, \dots, x_k are ordered from left to right, the search for the interval in which f_i is located can start from the interval in which f_{i-1} is located. Hence each x_i will be considered once. Thus the total running time is $O(m + k)$. We now show that the sorted list of distances between the facilities and the uncertain point can be constructed in $O(m + k)$ time. Since $E[d(p, C)]$ is a unimodal function, there is a facility f_i such that $E[d(f_j, C)]$ is decreasing for $f_j \leq f_i$ and increasing for $f_j \geq f_i$. Thus we have a

descending list of expected distances for f_1, f_2, \dots, f_i and ascending list of expected distances for $f_{i+1}, f_{i+2}, \dots, f_m$. We can merge these two lists into an ascending list of expected distances in $O(m)$ time. *Q.E.D.*

Corollary 4.2.4 *Let \mathcal{C} be a set of n uncertain points on a line each of which is specified by a piece-wise uniform function consisting of $k + 1$ pieces, and $F = \{f_1, f_2, \dots, f_m\}$ be a set of m facilities on the line. The expected distances between all pair of uncertain points and facilities can be computed and sorted in $O(nk + mn \log n)$ time.*

Proof. By Lemma 4.2.3, we can compute n sorted list of expected distances between customers and facilities in $O(nk + mn)$ time. The n sorted lists can be merged into a single list using min-heap in $O(mn \log n)$ time. *Q.E.D.*

We now consider the decision version of the uncertain r -gathering problem on a line. In order to solve the uncertain r -gathering problem on a line, we first solve the decision version and then search for the optimal solution. Given a set of uncertain customers \mathcal{C} , a set of facilities F on a line, and a number b , the decision uncertain r -gathering problem asks to determine whether there is an r -gathering A of \mathcal{C} to F such that $E[d(C, A(C))] \leq b$ for each $C \in \mathcal{C}$. The following lemma is known [43].

Lemma 4.2.5 ([43]) *Let C be an uncertain point on a line which is specified by a piece-wise uniform function consisting of $k + 1$ pieces and b is a number. Then the points p for which $E[d(C, p)] \leq b$ holds form an interval on the line.*

We call the interval which admits $E[d(C, p)] \leq b$ for customer C a (C, b) -interval and denote the interval by $[s_b(C), t_b(C)]$. Thus for each customer C we get a (C, b) -interval. Furthermore in any r -gathering A admitting $E[d(C, p)] \leq b$, $A(C)$ is lying between $[s_b(C), t_b(C)]$. Thus to find whether there is an r -gathering satisfying $E[d(C, p)] \leq b$ for each customer C , it is sufficient to solve the *interval r -gathering problem* which is defined as follows. Given a set of facilities F on a line and a set of customers \mathcal{C} where each customer $C \in \mathcal{C}$ has an interval $[s(C), t(C)]$ on

the line, the *interval r -gathering problem* asks to determine whether there is an r -gathering A such that each facility $f \in F$ serves zero or at least r customers and for each customer $C \in \mathcal{C}$, $s(C) \leq A(C) \leq t(C)$ holds.

We now give an algorithm for the interval r -gathering problem. Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers on a line where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$. An interval I_i is called the *leftmost interval* if for each $C_j \neq C_i$, $t(C_i) \leq t(C_j)$ holds, and the customer C_i is called the *leftmost customer*. A facility f_u is called the *preceding facility* of I_i if $s(C_i) \leq f_u \leq t(C_i)$ and there is no facility f'_u such that $f_u < f'_u \leq t(C_i)$. Similarly a facility f_u is called the *following facility* of I_i if $s(C_i) \leq f_u \leq t(C_i)$ and there is no facility f'_u such that $s(C_i) < f'_u \leq f_u$. We denote the set of right neighbors of C_i by N_i . We denote by \mathcal{C}_u the set of customers who have f_u contained within their intervals. We call a customer C_j a right neighbor of C_i if $t(C_j) \geq t(C_i)$ and $s(C_j) \leq t(C_i)$. We now have the following lemma.

Lemma 4.2.6 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers on a line where each customer C_i has an interval I_i . Let I_i be the leftmost interval, f_u be the preceding facility of I_i , and \mathcal{C}_u be the set of customers containing f_u in their intervals. If there is an interval r -gathering of \mathcal{C} to F , then there is an interval r -gathering with f_u be the leftmost open facility. Furthermore, the customers assigned to f_u have consecutive right end-points in \mathcal{C}_u including C_i .*

Proof. We first prove that there is an interval r -gathering with f_u be the leftmost open facility. Assume for a contradiction that there is no interval r -gathering with f_u be the leftmost open facility. Let A be an interval r -gathering with f_v be the leftmost open facility. We can observe that $f_v \leq f_u$, since in each interval r -gathering C_i is assigned to a facility within the interval I_i and f_u is the preceding facility of I_i . Let \mathcal{C}'_v be the set of customers assigned to f_v in A . For any customer C_j in \mathcal{C}'_v , we have $s(C_j) \leq f_v \leq f_u \leq t(C_i) \leq t(C_j)$, since I_i is the leftmost interval. We now can derive a new interval r -gathering by reassigning the customers \mathcal{C}'_v to f_u . A contradiction.

We now prove that the customers are assigned to f_u have consecutive right end-points in \mathcal{C}_u . We call a pair $C_j, C_k \in \mathcal{C}_u$ a reverse pair if $t(C_j) < t(C_k)$, C_k assigned to f_u , and C_j assigned to $f_v > f_u$. Assume for a contradiction that there is no interval r -gathering where the customers assigned to f_u have consecutive right end-points in \mathcal{C}_u . Let A' be an interval r -gathering with minimum number of reverse pairs but the number is not zero. Let C_j, C_k be a reverse pair in A' where $t(C_j) < t(C_k)$, and C_j is assigned to facility f_w , and C_k is assigned to f_u . Since $t(C_k) > t(C_j)$ and $f_w \geq f_u$, we get $s(C_k) \leq f_w \leq t(C_k)$. We now derive a new interval r -gathering with less reverse pairs by reassigning C_j to f_u and C_k to f_w . A contradiction.

Q.E.D.

We now have the following lemma.

Lemma 4.2.7 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers on a line where each customer C_i has an interval I_i . Let C_i be the leftmost customer, f_u be the preceding facility of I_i , and \mathcal{C}_u be the set of customers containing f_u in their intervals. Let C_j be the leftmost customer in $\mathcal{C} \setminus \mathcal{C}_u$, and $\mathcal{C}'_u \subseteq \mathcal{C}_u$ be the customers such that for each $C \in \mathcal{C}'_u$, $t(C) < t(C_j)$. Now if there is an interval r -gathering, then there is an interval r -gathering satisfying one of the following.*

- (a) *If $|\mathcal{C}'_u| < r$, then exactly the r leftmost customers in \mathcal{C}_u are assigned to f_u .*
- (b) *If $|\mathcal{C}'_u| \geq r$, then $\max\{|\mathcal{C}'_u| - r + 1, r\}$ leftmost customers of \mathcal{C}'_u are assigned to f_u .*

Proof.

(a) By Lemma 4.2.6, the customers assigned to f_u are consecutive in \mathcal{C}_u . Thus the leftmost r customers \mathcal{C}_u^l in \mathcal{C}_u are assigned to f_u . We now prove that there is an interval r -gathering where no customer in $\mathcal{C}_u \setminus \mathcal{C}_u^l$ is assigned to f_u . Assume for a contradiction that in every interval r -gathering there are some customers in $\mathcal{C}_u \setminus \mathcal{C}_u^l$ which are assigned to f_u . Let A be an interval r -gathering where the number of customers in $\mathcal{C}_u \setminus \mathcal{C}_u^l$ assigned to f_u is minimum, and C_k be a customer in $\mathcal{C}_u \setminus \mathcal{C}_u^l$ which is assigned to f_u . Since $|\mathcal{C}'_u| < r$, we get $t(C_k) > t(C_j)$. Let C_j is assigned to f_v in A . We now derive a new r -gathering by reassigning C_k to f_v , a contradiction.

(b) We first consider $r \leq |\mathcal{C}'_u| < 2r$. In this case $\max\{|\mathcal{C}'_u| - r + 1, r\} = r$. Since by Lemma 4.2.6 the customers assigned to f_u are consecutive in \mathcal{C}_u , the leftmost r customers in \mathcal{C}_u are assigned to f_u .

We now consider $|\mathcal{C}'_u| \geq 2r$. In this case, $\max\{|\mathcal{C}'_u| - r + 1, r\} = |\mathcal{C}'_u| - r + 1$. Let \mathcal{C}''_u be the leftmost $|\mathcal{C}'_u| - r + 1$ customers in \mathcal{C}'_u . Assume for a contradiction that there is no interval r -gathering where \mathcal{C}''_u are assigned to f_u . Let A' be an interval r -gathering with maximum number of customers $\mathcal{D}_u \subset \mathcal{C}''_u$ assigned to f_u . Let $C_s \in \mathcal{C}''_u$ be the customer with smallest $t(C_s)$ which is not assigned to f_u . Let C_s is assigned to $f_v \geq f_u$. By Lemma 4.2.6, any customer $C_t \in \mathcal{C}''_u$ with $t(C_t) \geq t(C_s)$ is not assigned to f_u . We first claim that the number of customers assigned to f_v is exactly r . Otherwise we can reassign C_s to f_u and thus contradicting our assumption. Let \mathcal{C}'_v be the customers assigned to f_v . We now claim that there is an interval r -gathering where \mathcal{C}'_v consists of r customers having consecutive right end-points in \mathcal{C}_u . Assume otherwise for a contradiction. Let A'' be an interval r -gathering with minimum number of crossing pair where a crossing pair is a pair of customer C_x, C_y with $t(C_x) \leq t(C_y)$, C_y assigned to f_v , C_x assigned to $f_w > f_v$. Since $t(C_x) \leq t(C_y)$ and $f_v \leq f_w$, we get $s(C_y) \leq f_w \leq t(C_y)$. We now derive a new interval r -gathering by reassigning C_x to f_v and C_y to f_w , a contradiction. Now since $|\mathcal{D}_u| < |\mathcal{C}'_u| - r + 1$, we get $|\mathcal{C}'_u \setminus \mathcal{D}_u| \geq r$. Thus $\mathcal{C}'_v \subset \mathcal{C}'_u$. We now derive a new interval r -gathering by assigning \mathcal{C}'_v to f_u . A contradiction. Q.E.D.

We now give an exact algorithm for the interval r -gathering problem.

We now have the following theorem.

Theorem 4.2.8 *The algorithm **Interval- r -gather** decides whether there is an interval r -gathering of \mathcal{C} to F , and constructs one if exists in $O(m + n \log n + nr^{\frac{n}{r}})$ time.*

Proof. The correctness of Algorithm **Interval- r -gather** is immediate from lemma 4.2.6 and 4.2.7.

We now estimate the running time of the algorithm. We can sort the customers based on their right end-points in $O(n \log n)$ time. For each customer we can precompute the preceding facility f in $O(n + m)$ time. For each facility f we can precompute the sets of customers \mathcal{C}_f containing each facility and the leftmost customer C' having left end-point on right of f in

Algorithm 4: Interval- r -gather(\mathcal{C}, F)

Input : A set \mathcal{C} of customers each having an interval and a set of F of facilities on a line

Output: An interval r -gathering if exists

if $|\mathcal{C}| < r$ or $F = \emptyset$ **then**

 | **return** \emptyset ;

endif

$C \leftarrow$ leftmost customer in \mathcal{C} ;

$f \leftarrow$ preceding facility of C ;

$\mathcal{C}_f \leftarrow$ customers containing f in their intervals;

$C' \leftarrow$ leftmost customer in $\mathcal{C} \setminus \mathcal{C}_f$;

$\mathcal{C}'_f \leftarrow$ customers in \mathcal{C}_f having smaller right end-point than $t(C')$;

$F' \leftarrow$ facilities right to f ;

if $|\mathcal{C}_f| < r$ **then**

 | **return** \emptyset ;

endif

if $|\mathcal{C}'_f| < r$ **then**

 | $\mathcal{D}_f \leftarrow r$ leftmost customers in \mathcal{C}_f ;

 | Assign \mathcal{D}_f to f ;

 | $Ans \leftarrow$ Interval- r -gather($\mathcal{C} \setminus \{\mathcal{D}_f\}, F'$);

 | **if** $Ans \neq \emptyset$ **then**

 | **Return** $Ans \cup \{\mathcal{D}_f\}$;

 | **endif**

 | **return** \emptyset ;

endif

$\mathcal{D}_f \leftarrow \max\{r, |\mathcal{C}'_f| - r + 1\}$ leftmost customers in \mathcal{C}_f ;

Assign \mathcal{D}_f to f ;

$\mathcal{C}''_f \leftarrow \mathcal{C}_f \setminus \{\mathcal{D}_f\}$;

while \mathcal{C}''_f is not empty **do**

 | $Ans \leftarrow$ Interval- r -gather($\mathcal{C} \setminus \{\mathcal{D}_f\}, F'$);

 | **if** $Ans \neq \emptyset$ **then**

 | **Return** $Ans \cup \{\mathcal{D}_f\}$;

 | **endif**

 | $C'' \leftarrow$ leftmost customer in \mathcal{C}''_f ;

 | Assign C'' to f ;

 | $\mathcal{D}_f \leftarrow \mathcal{D}_f \cup \{C''\}$;

end

return \emptyset ;

$O(n + m)$ time. In each call to Interval- r -gather, we need $O(|C_f|)$ time and at most r recursive calls to Interval- r -gather. Let $T(n)$ be the running time of the algorithm for n customers. We have $T(n) \leq O(|C_f|) + \sum_{i=1}^r T(n - r + 1) \leq O(nr^{\frac{n}{r}})$. Thus the running time of the algorithm is $O(m + n \log n + nr^{\frac{n}{r}})$. Q.E.D.

We now have the following theorem.

Theorem 4.2.9 *Let $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of uncertain customers on a line each of which is specified by a piece-wise uniform function consisting of $k + 1$ pieces, and $F = \{f_1, f_2, \dots, f_m\}$ be a set of m facilities on the line. Then the optimal r -gathering can be constructed in $O(nk + mn \log n + (m + n \log k + nr^{\frac{n}{r}}) \log mn)$ time.*

Proof. We give outline of an algorithm to compute optimal r -gathering. We first compute the $E[d(p, C_i)]$ function for each $C_i \in \mathcal{C}$. This takes $O(nk)$ time in total. By Corollary 4.2.4, we compute the sorted list of all expected distances between customers and facilities in $O(nk + mn \log n)$ time. We find the optimal r -gathering by binary search, using the algorithm for interval r -gathering $\log(mn)$ time. For solving each interval r -gathering problem, we compute the intervals for all customers in $O(n \log k)$ time and sort the intervals in $O(n \log n)$ time. Using the sorted list of intervals and facilities, we solve the interval r -gathering in $O(m + nr^{\frac{n}{r}})$. Thus finding optimal r -gathering by binary search requires $O(nk + mn \log n + (m + n \log k + nr^{\frac{n}{r}}) \log mn)$ time. Q.E.D.

4.2.2 Uniform Distribution

In this section we give an algorithm for uncertain r -gathering problem when each customer location is specified by a uniform distribution.

In the uniform distribution model, location of each customer C_i is specified by a uniform function $g_i : \mathbb{R} \rightarrow \mathbb{R}^+ \cup \{0\}$ where $g_i(p) = 1/(t_i - s_i)$ if $t_i \leq p \leq s_i$ and $g_i(p) = 0$ otherwise. We denote the uniform distribution between $[s_i, t_i]$ by $U(s_i, t_i)$. The customer C_i has a uniform distribution $U(s_i, t_i)$ is denoted by $C_i \sim U(s_i, t_i)$. Figure 4.2(a) illustrates a uniform distribution

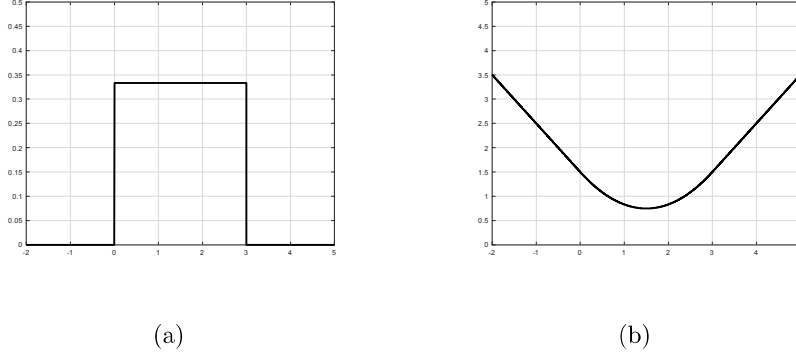


Figure 4.2: (a) Illustration of uniform distribution as a PDF and (b) corresponding function of expected distance.

where $s_i = 0$ and $t_i = 3$. The range of $U(s_i, t_i)$, denoted by l_i , is the value of $t_i - s_i$, and the mean of $U(s_i, t_i)$, denoted by μ_i , is the value of $\frac{s+t}{2}$. It is easy to observe that, the uniform distribution model is a special case of the histogram model described in Section 4.2.1. Thus by Lemma 4.2.1, 4.2.2 and 4.2.5, the function $E[d(p, C)]$ is a unimodal function where each piece is a parabola and the points for which $E[d(p, C)] \leq b$ form an interval. We now have the following lemma.

Lemma 4.2.10 *Let $C \sim U(s, t)$ be an uncertain point which is specified by a uniform function within $[s, t]$. Then the function $E[d(p, C)]$ consists of a parabola in the interval $[s, t]$ and two straight lines of slope +1 and -1 in interval (t, ∞) and $(-\infty, s)$ respectively. Furthermore the minimum value of $E[d(p, C)]$ is $\frac{1}{4}$ and the value of $E[d(p, C)]$ at s, t is $\frac{1}{2}$.*

Proof. We use the Equation 4.1 to compute the function $E[d(p, C)]$. For $p < s$, we get-

$$\begin{aligned} E[d(p, C)] &= -p + \frac{1}{2} \frac{1}{(t-s)} (t^2 - s^2) \\ &= -p + \frac{s+t}{2} \end{aligned}$$

Similarly for $p > t$, we get $E[d(p, C)] = p + \frac{s+t}{2}$. Thus the function $E[d(p, C)]$ is a straight line

of slope -1 for $p < s$, and a straight line of slope 1 for $p > t$. Now for $s \leq p \leq t$ we get,

$$\begin{aligned} E[d(p, C)] &= \frac{1}{t-s}p^2 - \frac{1}{t-s}(s+t)p + \frac{1}{2} \frac{1}{t-s}(s^2 + t^2) \\ &= \frac{1}{t-s} \left(p - \frac{s+t}{2} \right)^2 + \frac{t-s}{4} \end{aligned}$$

Thus $E[d(p, C)]$ is a parabola in the interval $[s, t]$. Hence we get-

$$E[d(p, C)] = \begin{cases} \mu - p & \text{if } p < s \\ \frac{1}{l} (p - \mu)^2 + \frac{l}{4} & \text{if } s \leq p \leq t \\ -\mu + p & \text{if } p > t \end{cases} \quad (4.2)$$

At $p = s$ we get $E[d(s, C)] = \frac{1}{t-s} \left(s - \frac{s+t}{2} \right)^2 + \frac{t-s}{4} = \frac{t-s}{2} = \frac{l}{2}$. Similarly, $E[d(t, C)] = \frac{l}{2}$. Now for $p < s$ and $p > t$, $E[d(p, C)] \geq \frac{t-s}{2}$. The minimum value of the parabola $\frac{1}{t-s} \left(p - \frac{s+t}{2} \right)^2 + \frac{t-s}{4}$ is at $p = \frac{s+t}{2}$, which is $\frac{l}{4}$. $\mathcal{Q.E.D.}$

By Equation 4.2, we can calculate $E[d(p, C)]$ for a fixed point p in $O(1)$ time. In the following lemma, we show that for a customer C and a number b we can compute the (C, b) -interval in $O(1)$ time.

Lemma 4.2.11 *Let $C \sim U(s, t)$ be an uncertain point and b be a number. Then the (C, b) -interval can be computed in $O(1)$ time.*

Proof. To find the (C, b) -interval, we first compute the inverse of the Equation 4.2. For $E[d(p, C)] = b \geq \frac{l}{2}$, we have $p < s$ or $p > t$. Thus we get, $p = \mu \pm b$. For $\frac{l}{4} \leq E[d(p, C)] = b \leq \frac{l}{2}$, we have $s \leq p \leq t$. Thus we get $p = \mu \pm \sqrt{l(b - \frac{l}{4})}$. Finally there is no p for which $E[d(p, C)] < \frac{l}{4}$. Hence the (C, b) -interval for $b < \frac{l}{4}$ is empty. Thus the (C, b) -interval I can be written as following.

$$I = \begin{cases} [\mu - b, \mu + b] & \text{if } b > \frac{l}{2} \\ [\mu - \sqrt{l(b - \frac{l}{4})}, \mu + \sqrt{l(b - \frac{l}{4})}] & \text{if } \frac{l}{4} \leq b \leq \frac{l}{2} \\ \emptyset & \text{if } b < \frac{l}{4} \end{cases} \quad (4.3)$$

By Equation 4.3 we can compute (C, b) -interval in $O(1)$ time.

$\mathcal{Q.E.D.}$

Let $C_i \sim U(s_i, t_i), C_j \sim U(s_j, t_j)$ be two uncertain points. Let $l_{max} = \max\{l_i, l_j\}$ and $l_{min} = \min\{l_i, l_j\}$. We call C_i, C_j *well-separated* if none of the intervals $[s_i, t_i]$ and $[s_j, t_j]$ is contained within the other and $|\mu_i - \mu_j| \geq \frac{1}{2}\sqrt{l_{min}(l_{max} - l_{min})}$.

Lemma 4.2.12 *Let $C_i \sim U(s_i, t_i), C_j \sim U(s_j, t_j)$ be two uncertain well-separated points and b be a number. Let I_i, I_j be the (C_i, b) -interval and (C_j, b) -interval respectively. Then none of I_i and I_j is contained within the other.*

Proof. Since C_i and C_j are well-separated, it is easy to observe that $s_i = s_j$ if and only if $t_i = t_j$. In this case the claim trivially holds. We thus consider otherwise. Without loss of generality we assume that, $s_i < s_j, t_i < t_j$ and $l_i \leq l_j$. Since $s_i < s_j$ and $t_i < t_j$, we get $\mu_i = \frac{s_i+t_i}{2} < \frac{s_j+t_j}{2} = \mu_j$. We now have two cases.

Case 1: $l_j \geq 2l_i$. In this case we have three subcases to consider.

Case 1a: $b > \frac{l_j}{2}$.

By Equation 4.3 we get $I_i = [\mu_i - b, \mu_i + b]$. Similarly, we get $I_j = [\mu_j - b, \mu_j + b]$. Now since $\mu_i < \mu_j$, we get $\mu_i - b < \mu_j - b$ and $\mu_i + b < \mu_j + b$. Thus none of I_i, I_j is contained within the other.

Case 1b: $\frac{l_j}{4} \leq b \leq \frac{l_j}{2}$.

Since $l_j \geq 2l_i$, we have $\frac{l_i}{2} \leq \frac{l_j}{4}$. By Equation 4.3, $I_i = [\mu_i - b, \mu_i + b]$ and $I_j = [\mu_j - \sqrt{l_j(b - l_j/4)}, \mu_j + \sqrt{l_j(b + l_j/4)}]$. Assume for a contradiction that either I_i or I_j is contained within the other. We first consider I_i is contained within I_j . Since $b > \frac{l_j}{2}$, we get $\mu_i - b < s_i$ and $\mu_i + b > t_i$. On the other hand, since $b \leq \frac{l_j}{2}$ we get $\mu_j - \sqrt{l_j(b - l_j/4)} \geq s_j$ and $\mu_j - \sqrt{l_j(b - l_j/4)} \leq t_j$. Now since I_i is contained within I_j , we have $\mu_j - \sqrt{l_j(b - l_j/4)} \leq \mu_i - b$ and $\mu_j + \sqrt{l_j(b - l_j/4)} \geq \mu_i + b$. Thus we get $s_j < s_i$ and $t_i < t_j$, a contradiction.

We now consider I_j is contained within I_i . In this case, $\mu_i - b \leq \mu_j - \sqrt{l_j(b - l_j/4)}$ and $\mu_i + b \geq \mu_j + \sqrt{l_j(b - l_j/4)}$. Note that, the absolute value of the slope of tangent of parabola $\frac{1}{l_j}(p - \mu_j)^2 + \frac{l_j}{4}$ at any point $p \in [s_j, t_j]$ is less than 1. Hence the interval I_j at $b = \frac{l_j}{4}$ must be contained within the interval I_i at $b = \frac{l_j}{4}$. At $b = \frac{l_j}{4}$, we have $I_i = [\mu_i - \frac{l_j}{4}, \mu_i + \frac{l_j}{4}]$ and $I_j = [\mu_j, \mu_j]$. Since I_j is contained within I_i , we get $\mu_j - \mu_i \leq \frac{l_j}{4}$. Now since I_i is not contained

within I_j and $l_j \geq 2l_i$, we get-

$$\begin{aligned}
s_j &= \mu_j - \frac{l_j}{2} \\
&\leq \mu_i + \frac{l_j}{4} - \frac{l_i}{2} \\
&= \mu_i - \frac{l_i}{4} \\
&\leq \mu_i - \frac{l_i}{2} \\
&= s_i
\end{aligned}$$

Similarly, we can show $t_j \geq t_i$. Thus I_i is contained within I_j , a contradiction.

Case 1c: $b < \frac{l_j}{4}$.

Since $b < \frac{l_j}{4}$, we get $I_j = \emptyset$. Thus the claim holds.

Case 2: $l_j < 2l_i$. For this case, we have four subcases to consider.

Case 2a: $b > \frac{l_j}{2}$. Similar to Case 1a.

Case 2b: $\frac{l_i}{2} < b \leq \frac{l_j}{2}$.

In this case by Equation 4.3, $I_i = [\mu_i - b, \mu_i + b]$ and $I_j = [\mu_j - \sqrt{l_j(b - l_j/4)}, \mu_j + \sqrt{l_j(b + l_j/4)}]$.

Assume for a contradiction that either I_i or I_j is contained within the other. We first consider

I_i is contained within I_j . Since $b > \frac{l_i}{2}$, we get $\mu_i - b < s_i$ and $\mu_i + b > t_i$. On the other

hand, since $b \leq \frac{l_j}{2}$ we get $\mu_j - \sqrt{l_j(b - l_j/4)} \geq s_j$ and $\mu_j - \sqrt{l_j(b - l_j/4)} \leq t_j$. Now since I_i is

contained within I_j , we have $\mu_j - \sqrt{l_j(b - l_j/4)} \leq \mu_i - b$ and $\mu_j + \sqrt{l_j(b - l_j/4)} \geq \mu_i + b$. Thus

we get $s_j < s_i$ and $t_i < t_j$, a contradiction.

We now consider I_j is contained within I_i . In this case, $\mu_i - b \leq \mu_j - \sqrt{l_j(b - l_j/4)}$ and $\mu_i + b \geq$

$\mu_j + \sqrt{l_j(b - l_j/4)}$. Since the absolute value of the slope of tangent of parabola $\frac{1}{l_j}(p - \mu_j)^2 + \frac{l_j}{4}$ at

any point $p \in [s_j, t_j]$ is less than 1, the interval I_j at $b = \frac{l_i}{2}$ must be contained within the interval

I_i at $b = \frac{l_i}{2}$. At $b = \frac{l_i}{4}$, we have $I_i = [\mu_i - \frac{l_i}{2}, \mu_i + \frac{l_i}{2}]$ and $I_j = [\mu_j - \sqrt{l_j(\frac{l_i}{2} - \frac{l_j}{4})}, \mu_j + \sqrt{l_j(\frac{l_i}{2} - \frac{l_j}{4})}]$.

Thus I_j is contained within I_i if and only if $\mu_j - \mu_i \leq \frac{l_i}{2} - \frac{1}{2}\sqrt{l_j(\frac{l_i}{2} - \frac{l_j}{4})}$. Now since $l_i \leq l_j$, we

have $2l_i - l_j \leq l_j$. Hence we get,

$$\begin{aligned} \frac{l_i}{2} - \frac{1}{2}\sqrt{l_j\left(\frac{l_i}{2} - \frac{l_j}{4}\right)} &\leq \frac{l_i}{2} - \frac{1}{2}\sqrt{\left(\frac{l_i}{2} - \frac{l_j}{4}\right)^2} \\ &= \frac{l_i}{2} - l_i + \frac{l_j}{2} \\ &= \frac{l_j - l_i}{2} \end{aligned}$$

Thus I_j is contained within I_i if and only if $\mu_j - \mu_i \leq \frac{l_j - l_i}{2}$. Now since, none of I_i, I_j is contained within the other, we have $\mu_j - \mu_i > \frac{l_j - l_i}{2}$.

Case 2c: $\frac{l_j}{4} \leq b \leq \frac{l_i}{2}$. In this case, $I_i = [\mu_i - \sqrt{l_i(b - l_i/4)}, \mu_i + \sqrt{l_i(b + l_i/4)}]$ and $I_j = [\mu_j - \sqrt{l_j(b - l_j/4)}, \mu_j + \sqrt{l_j(b + l_j/4)}]$. Assume for a contradiction that I_i or I_j is contained within the other. We first consider I_i is contained within I_j . Since $\mu_i \leq \mu_j$, I_i is contained within I_j if and only if $\mu_i - \sqrt{l_i(b - l_i/4)} \geq \mu_j - \sqrt{l_j(b - l_j/4)}$ which yields $\mu_j - \mu_i \leq \sqrt{l_j(b - l_j/4)} - \sqrt{l_i(b - l_i/4)}$. Similarly, I_j is contained within I_i if and only if $\mu_j - \mu_i \leq \sqrt{l_i(b - l_i/4)} - \sqrt{l_j(b - l_j/4)}$. Thus either I_i or I_j is contained within the other if and only if $\mu_j - \mu_i \leq |\sqrt{l_j(b - l_j/4)} - \sqrt{l_i(b - l_i/4)}|$.

Let $h(b) = \sqrt{l_j(b - l_j/4)} - \sqrt{l_i(b - l_i/4)}$. We now show that, the function $h(b)$ is increasing at any point $b \geq l_j/4$. Clearly, $h(b)$ is not defined for $b < l_j/4$. We can calculate the derivative of $h(b)$ as follows.

$$\begin{aligned} \frac{d}{db}h(b) &= \sqrt{\frac{l_j}{4b - l_j}} - \sqrt{\frac{l_i}{4b - l_i}} \\ &= \frac{\sqrt{l_j(4b - l_i)} - \sqrt{l_i(4b - l_j)}}{\sqrt{(4b - l_j)(4b - l_i)}} \end{aligned}$$

Since $l_i \leq l_j$, we get $\sqrt{l_j(4b - l_i)} \geq \sqrt{l_i(4b - l_j)}$. Thus $\frac{d}{db}h(b) > 0$ for any $b \geq l_j/4$, and hence the function $h(b)$ is increasing. We now show that the maximum value of $|h(b)|$ within interval $[\frac{l_j}{4}, \frac{l_i}{2}]$ is at $b = l_j/4$. We first observe that $h(b) = 0$ at $b = \frac{l_i + l_j}{4}$. Since $l_j \geq l_i$, $\frac{l_i + l_j}{4} \geq \frac{l_i}{2}$. Thus $|h(b)|$ is decreasing in the interval $[\frac{l_j}{4}, \frac{l_i}{2}]$. Hence the maximum value of $|h(b)|$ within the interval $[\frac{l_j}{4}, \frac{l_i}{2}]$ is at $b = \frac{l_j}{4}$. Thus the maximum value of $|h(b)|$ is

$$\begin{aligned} \left| h\left(\frac{l_j}{4}\right) \right| &= \left| \sqrt{l_j\left(\frac{l_j}{4} - \frac{l_j}{4}\right)} - \sqrt{l_i\left(\frac{l_j}{4} - \frac{l_i}{4}\right)} \right| \\ &= \frac{1}{2}\sqrt{l_i(l_j - l_i)} \end{aligned}$$

Since C_i, C_j are well-separated, $\mu_j - \mu_i$ cannot be greater than $\frac{1}{2}\sqrt{l_i(l_j - l_i)}$, a contradiction.

Case 2d: $b < \frac{l_j}{4}$. Similar to Case 1c.

Q.E.D.

We now give an algorithm for the decision version of uncertain r -gathering problem when the customer locations are specified by well-separated uniform distributions. Clearly, the decision version is the interval r -gathering problem as specified in Section 4.2.1. Since the intervals are proper for the well-separated uniform distribution, we call the decision version *proper interval r -gathering problem*. The proper interval r -gathering problem is defined as follows. Given a set of facilities $F = \{f_1, f_2, \dots, f_m\}$ on a line and a set of customers $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ on the line such that I_i is not contained within any interval $I_j \neq I_i$, the *interval r -gathering problem* asks to determine whether there is an r -gathering A such that each facility f_j serves zero or at least r customers and for each customer C_i , $s(C_i) \leq A(C_i) \leq t(C_i)$ holds. We assume the customers are ordered in increasing order of right end-points and the facilities are ordered from left to right.

If all the intervals are proper, we can improve the running time of Algorithm Interval- r -gather by dynamic programming approach. We memoize the call to the interval r -gather. Since all the intervals are proper, each call to Interval- r -gather with C_i as the leftmost customer has same set of customers. Thus we have at most n distinct subproblems, and in each call to Interval- r -gather we take $O(n)$ time other than recursive calls. Hence with memoization, the algorithm runs in $O(m + n^2)$ time. However we can further improve the running time by modifying the algorithm for (k, r) -gathering on a line by Akagi and Nakano [9]. We solve the proper interval r -gathering problem by solving the problem $P(i)$ which is defined as follows. Given a set of facilities $F = \{f_1, f_2, \dots, f_m\}$ on a line and customers $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ where each customer has an interval $I_i = [s(C_i), t(C_i)]$ such that I_i is not contained within any interval I_j , the problem $P(i)$ asks to find a set of customers \mathcal{C}_i and an interval r -gathering

A of customers $\mathcal{C}_i \subseteq \mathcal{C}$ to $F_i = \{f_1, f_2, \dots, f_i\}$ such that (1) \mathcal{C}_i contains all customers with $t(C_i) \leq f_i$, (2) f_i has r customers, and (3) $\max_{C \in \mathcal{C}_i} \{t(C)\}$ is minimum. We denote the customer with $\max_{C \in \mathcal{C}_i} \{t(C)\}$ by $z(i)$. Let C_j be the customer such that $s(C_j)$ is the minimum. We can observe that there is a proper interval r -gathering if and only if $P(i)$ has a solution for $f_i \geq s(C_j)$. We have the following lemma.

Lemma 4.2.13 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. If $P(i)$ has a solution, then there is an interval r -gathering where customers assigned to each open facility has consecutive right end-points.*

Proof. In an interval r -gathering A we call a pair of customers C_u, C_v a crossing pair, if $t(C_u) < t(C_v)$ and $A(C_u) \geq A(C_v)$. Let \mathcal{C}_i be the set of customers corresponding to the solution of $P(i)$. For a contradiction, assume there is no interval r -gathering where customers assigned to each open facility is consecutive. Let A_i be an interval r -gathering corresponding to $P(i)$ with minimum number of crossing pairs. Let C_u, C_v be a crossing pair. Since all the intervals are proper, $s(C_u) < s(C_v)$. Thus we have $s(C_u) \leq A_i(C_v) \leq t(C_u)$, and $s(C_v) \leq A_i(C_u) \leq t(C_v)$. Now we can derive a new r -gathering by reassigning C_u to $A_i(C_v)$ and C_v to $A_i(C_u)$, which reduces the number of crossing pairs by one. A contradiction. *Q.E.D.*

We now have the following lemma.

Lemma 4.2.14 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. Let f_i, f_j be two facilities with $f_i < f_j$. If both $P(i)$ and $P(j)$ has solution then $t(z(i)) \leq t(z(j))$.*

Proof. For a contradiction assume $t(z(i)) > t(z(j))$. Let A_j be an interval r -gathering corresponding to $P(j)$. Since all the intervals are proper, we have $s(z(i)) > s(z(j))$. We have $s(z(j)) \leq f_i$, since $z(j)$ is assigned to f_j in A_j . Let \mathcal{C}'_j be the set of customers assigned to any

facility between f_i to f_j (including f_j) in A_j . For any customer $C_k \in \mathcal{C}'_j$, we have $s(C_k) \leq f_i$ and $t(C_k) \geq f_j$. We now derive a new interval r -gathering A'_j by reassigning the leftmost r customers \mathcal{C}'_j to f_i . Clearly, $\max_{C \in \mathcal{C}'_j} \{t(C)\} < t(z(i))$ and thus A'_j is a solution of $P(i)$, a contradiction. Q.E.D.

Using Lemma 4.2.13 and 4.2.14, we can determine whether $P(i)$ has solution by the following way. If $f_i \leq t(C_1)$, then $P(i)$ has solution if and only if f_i is contained within at least r intervals. Otherwise if $f_i > t(C_1)$, then there must be two or more open facilities. In this case $P(i)$ has a solution if and only if there is a facility $f_{i'}$ such that $P(i')$ has a solution and satisfying one of the following conditions.

co1: If $t(z(i')) \geq f_i$, then there are at least r customers in $\mathcal{C} \setminus \mathcal{C}_{i'}$ containing f_i .

co2: If $t(z(i')) < f_i$, then there are at least r customers in $\mathcal{C} \setminus \mathcal{C}_{i'}$ which contain f_i and each customer $C_j \in \mathcal{C} \setminus \mathcal{C}_{i'}$ having $t(C_j) < f_i$ has $s(C_j) \leq f_{i'}$.

Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. Let $f_i, f_{i'}, f_{i''}$ be three facilities such that $f_{i''} \leq f_{i'} \leq f_i$ and each of $P(i), P(i'), P(i'')$ has a solution. Let $\mathcal{C}_{i''}$ be the set of customers such that there is an interval r -gathering of $\mathcal{C}_{i''}$ to F_i with f_i open and $f_{i''}$ be the second rightmost open facility such that $\max_{C \in \mathcal{C}_{i''}} \{t(C)\}$ is minimum. Let $\mathcal{C}_{i'}$ be the set of customers such that there is an interval r -gathering of $\mathcal{C}_{i'}$ to F_i with f_i open and $f_{i'}$ be the second rightmost open facility such that $\max_{C \in \mathcal{C}_{i'}} \{t(C)\}$ is minimum. Then $\max_{C \in \mathcal{C}_{i''}} \{t(C)\} \leq \max_{C \in \mathcal{C}_{i'}} \{t(C)\}$.

Let $P(i)$ has a solution and $f_i > t(C_1)$. For such $P(i)$, we have one or more open facilities. We call the minimum second rightmost open facility $f_{i'}$ for which $P(i)$ has a solution as the mate of f_i , denoted by $mate(f_i)$. We have the following lemma.

Lemma 4.2.15 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. Let $f_i \neq f_n$ be a facility such that $f_i > t(C_1)$ and each of $P(i), P(i+1)$ has a solution. Then $mate(f_i) \leq mate(f_{i+1})$.*

Proof. For a contradiction assume $\text{mate}(f_i) > \text{mate}(f_{i+1})$. Let $f_j = \text{mate}(f_i)$ and $f_{j'} = \text{mate}(f_{i+1})$. By Lemma 4.2.14 we have $t(z(j)) \geq t(z(j'))$. Since $f_{j'}$ is mate of f_{i+1} , there is no customer C such that $f_{j'} < s(C) \leq t(C) < f_{i+1}$. If $t(z(j)) < f_i$, then $f_{j'}$ is also a mate of f_j , a contradiction. Now if $t(z(j)) \geq f_j$, then $f_{j'}$ is a mate of f_j since $t(z(j')) \leq t(z(j))$, a contradiction. $\quad \text{Q.E.D.}$

We now have the following lemma.

Lemma 4.2.16 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. Let f_i be a facility such that $f_i > t(C_1)$ and $f_{i'}$ be the leftmost facility such that $P(i')$ has solution, there is no customer C_k with $f_{i'} < s(C_k) \leq t(C_k) < f_i$, and $\mathcal{C} \setminus \mathcal{C}_{i'}$ has less than r customers containing f_i . Then the following holds.*

(1) *No facility f_j with $f_j \geq f_{i'}$ is a mate of f_i .*

(2) *If $P(i)$ has no solution and $P(i+1)$ has a solution, then $\text{mate}(f_{i+1}) \geq f_{i'}$.*

Proof.

(1) By Lemma 4.2.14 for any facility $f_j \geq f_{i'}$, if $P(j)$ has a solution, then $\text{mate}(j) \geq \text{mate}(i')$. Thus the number of customers in $\mathcal{C} \setminus \mathcal{C}_j$ containing f_i in their interval is less than r .

(2) Assume for a contradiction that $\text{mate}(f_{i+1}) \leq f_{i'}$. Let $f_{i''} = \text{mate}(f_{i+1})$. We have $t(z(i'')) \leq t(z(i'))$. Since $f_{i''}$ is a mate of f_{i+1} , there is no customer C_k with $f_{i''} < s(C_k) \leq t(C_k) < f_{i+1}$. Thus there is no customer C_k such that $f_{i''} < s(C_k) \leq t(C_k) < f_i$. Since there are less than r customers containing f_i in $\mathcal{C} \setminus \mathcal{C}_{i'}$ and $t(z(i'')) \leq t(z(i'))$, there are also less than r customers containing f_i in $\mathcal{C} \setminus \mathcal{C}_{i''}$. Hence $f_{i'}$ cannot be the leftmost facility such that $P(i')$ has solution, there is no customer C_k with $f_{i'} < s(C_k) \leq t(C_k) < f_i$, and $\mathcal{C} \setminus \mathcal{C}_{i'}$. A contradiction. $\quad \text{Q.E.D.}$

By Lemma 4.2.15 and 4.2.16, we observe that we can search for $\text{mate}(f_{i+1})$ from where the search for mate of $\text{mate}(f_i)$ ends. Based on these observation we give the following Algorithm called proper interval r -gather.

If the intervals are sorted according to their right end-points and the facilities are ordered from left to right, then we can preprocess the set of customers containing each facility in linear time. Each customer and each facility have to be processed for a constant number of times. Hence the algorithm runs in $O(n + m)$ time. We thus have the following theorem.

Theorem 4.2.17 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has an interval $I_i = [s(C_i), t(C_i)]$ and no interval is contained within any other. The algorithm **Proper-interval- r -gather** decides whether there is an interval r -gathering of \mathcal{C} to F , and constructs one if exists in $O(n + m)$ time.*

We now give outline of the algorithm to solve uncertain r -gathering problem on a line where the customer locations are specified by well-separated uniform distributions. Computing the $E[d(p, C_i)]$ for all the customers takes $O(n)$ time. Computing the distances between each pair of customers and facilities takes $O(nm)$ time and the sorting of distances requires $O(mn \log(mn))$ time. We do binary search on the ordered list of distances to find the optimal r -gathering. We can compute the (C, b) -intervals for all customers in $O(n)$ time. The (C, b) -intervals can be sorted in $O(n \log n)$ time. Solving each decision instance takes $O(n + m)$ time. Thus to find the optimal solution by binary search we need $O((n \log n + m) \log(mn))$. Hence the running time is $O(mn \log(mn) + n \log n \log(mn))$. Thus we have the following theorem.

Theorem 4.2.18 *Let $F = \{f_1, f_2, \dots, f_m\}$ be a set of facilities on a line and $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$ be a set of customers where each customer C_i has a well-separated uniform distribution. Then an optimal r -gathering of \mathcal{C} to F can be constructed in $O(mn \log n + (m + n \log n) \log(mn))$ time.*

4.3 Summary

In this chapter, we give algorithms for the uncertain r -gathering problem on a line for two input probability distributions. For the general case where the customer locations are specified

by histograms, our algorithm runs in exponential time. We also give an $O(mn \log n + (m + n \log n) \log(mn))$ time algorithm for uncertain r -gathering problem when the customer locations are given in well-separated uniform distributions.

Algorithm 5: Proper-interval- r -gather(\mathcal{C}, F)

Input : A set \mathcal{C} of customers each having an interval where no interval is contained within other and a set of F of facilities on a line

Output: An interval r -gathering if exists

if $|\mathcal{C}| < r$ *or* $F = \emptyset$ **then**

 | **return** \emptyset ;

endif

$i \leftarrow 1$;

while $f_i \leq t(C_1)$ **do**

 | **if** $f_i \geq s(C_r)$ **then**

 | $z(i) \leftarrow C_r$;

 | **endif**

 | $i \leftarrow i + 1$;

end

$j \leftarrow 1$;

while $i \leq m$ **do**

 | $\mathcal{C}_{f_i} \leftarrow$ set of customers containing f_i ;

 | **while** $j \leq i$ **do**

 | **if** $t(z(j)) < f_i$ *and* $|\mathcal{C}_{f_i}| \geq r$ *and there is no customer* C *with*

 | $f_j < s(C) \leq t(C) < f_i$ **then**

 | $C \leftarrow r$ -th customer in \mathcal{C}_{f_i} ;

 | $z(i) \leftarrow C$;

 | $mate(i) \leftarrow j$;

 | **break**;

 | **endif**

 | $\mathcal{C}' \leftarrow$ customers C with $t(C) > t(z(i))$;

 | **if** $t(z(j)) \geq f_i$ *and there are at least* r *customers in* $\mathcal{C}_{f_i} \cap \mathcal{C}'$ **then**

 | $C \leftarrow r$ -th customer in $\mathcal{C}_{f_i} \cap \mathcal{C}'$;

 | $z(i) \leftarrow C$;

 | $mate(i) \leftarrow j$;

 | **break**;

 | **endif**

 | **if** *There are no customer between* f_j *and* f_i , *and there are less than* r *customers in* $\mathcal{C}_{f_i} \cap \mathcal{C}'$ **then**

 | **break**;

 | **endif**

 | $j \leftarrow j + 1$;

 | **end**

 | $i \leftarrow i + 1$;

end

if $P(i)$ *has solution for* $f_i \geq s(C_n)$ **then**

 | $A_i \leftarrow$ Assignment of customers in open facility in $P(i)$;

 | **return** A_i ;

endif

return \emptyset ;

Chapter 5

Conclusion

In this chapter we review the ideas discussed in the previous chapters and explain some interesting open problems in this field.

In this thesis we have dealt with two problems called the r -gathering problem and the r -gather clustering problem. At first, in Chapter 1 we have defined the two problems and discussed their applications. We have also reviewed the literature of the two problems and give an objective of this thesis.

In Chapter 2 we have given some preliminary definitions on graph, algorithmic theory and probability theory. We have also discussed about different graph classes that are necessary to understand this thesis work.

In Chapter 3 we have considered the r -gathering problem and the r -gather clustering problems when the customers and facilities are lying on a star. We give algorithm for both problems which run in polynomial time if the number of rays of the star is constant. We also showed a relevant variant of r -gathering problem on star is NP-hard.

In Chapter 4 we have considered the r -gathering problem when the customer locations are uncertain. For one-dimensional case, we give an exact exponential algorithm for solving the problem when customer locations are specified by histogram. For more restricted case when the customer locations are specified by well-separated uniform distribution, we give a polynomial time algorithm for the problem.

In this thesis we have tried add more results on the r -gathering and the r -gather clustering problem. However he following problems are still open and remained as future works.

1. Improve the running time of the r -gathering and r -gather clustering problems on star.
2. Determine the complexity of the problems when the customers and facilities are lying on a tree.
3. Find a polynomial time algorithm for the uncertain r -gathering problem when the customer and facilities are on a line and the customer locations are specified by histograms.
4. Determine the complexity of the uncertain r -gathering problem on a line for some other probability distributions.
5. Find an approximation algorithm for uncertan r -gathering problem on plane.

List of Publications

1. Shareef Ahmed, Shin-ichi Nakano and Md. Saidur Rahman, r -Gatherings on a Star, In *Proceedings of WALCOM: Algorithms and Computation*, volume 11355 of Lecture Notes in Computer Science, pages 31–42, 2019, Springer Nature Switzerland.

References

- [1] F. N. Abu-Khzam, C. Bazgan, K. Casel, and H. Fernau. Clustering with lower-bounded sizes. *Algorithmica*, 80(9):2517–2550, Sep 2018.
- [2] P. K. Agarwal, S. Cheng, Y. Tao, and K. Yi. Indexing uncertain data. In *Proceedings of the Twenty-Eighth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009*, pages 137–146, 2009.
- [3] P. K. Agarwal, A. Efrat, S. Sankararaman, and W. Zhang. Nearest-neighbor searching under uncertainty I. *Discrete & Computational Geometry*, 58(3):705–745, 2017.
- [4] P. K. Agarwal, S. Har-Peled, S. Suri, H. Yildiz, and W. Zhang. Convex hulls under uncertainty. *Algorithmica*, 79(2):340–367, 2017.
- [5] P. K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Computing Surveys*, 30(4):412–458, 1998.
- [6] G. Aggarwal, R. Panigrahy, T. Feder, D. Thomas, K. Kenthapadi, S. Khuller, and A. Zhu. Achieving anonymity via clustering. *ACM Transactions on Algorithms*, 6(3):49:1–49:19, 2010.
- [7] S. Ahmadian and C. Swamy. Improved approximation guarantees for lower-bounded facility location. In *Approximation and Online Algorithms*, pages 257–271, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

- [8] S. Ahmadian and C. Swamy. Approximation algorithms for clustering problems with lower bounds and outliers. In *Proceedings of 43rd International Colloquium on Automata, Languages, and Programming, ICALP*, pages 69:1–69:15, 2016.
- [9] T. Akagi and S. Nakano. On r -gatherings on the line. In *Proceedings of Frontiers in Algorithmics*, volume 9130 of Lecture Notes in Computer Science, pages 25–32, Cham, 2015. Springer International Publishing.
- [10] A. Armon. On min-max r -gatherings. *Theoretical Computer Science*, 412(7):573 – 582, 2011.
- [11] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k -median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004.
- [12] C.-Y. Chow and M. F. Mokbel. Trajectory privacy in location-based services and data publication. *ACM SIGKDD Explorations Newsletter*, 13(1):19–29, 2011.
- [13] G. Cornuejols, G. L. Nemhauser, and L. A. Wolse. The uncapacitated facility location problem. *Discrete Location Theor*, 222:45–58, 2013.
- [14] S. Dasgupta, C. H. Papadimitriou, and U. Vazirani. *Algorithms*. McGraw-Hill, 2006.
- [15] Z. Drezner and H. W. Hamacher. *Facility Location: Applications and Theory*. Springer, New York, 2004.
- [16] Z. Drezner and G. O. Wesolowsky. Single facility l_p -distance minimax location. *SIAM Journal on Matrix Analysis and Applications*, 1(3):315–321, 1980.
- [17] R. J. Fowler, M. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, 12(3):133–137, 1981.
- [18] G. N. Frederickson and D. B. Johnson. Generalized selection and ranking: Sorted matrices. *SIAM Journal on Computing*, 13(1):14–30, 1984.

- [19] H. N. Gabow and R. E. Tarjan. Algorithms for two bottleneck optimization problems. *Journal of Algorithms*, 9(3):411 – 417, 1988.
- [20] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.
- [21] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.
- [22] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 603–612, 2000.
- [23] Y. Han and S. Nakano. On r -gatherings on the line. In *Proceedings of FCS 2016*, pages 99–104, 2016.
- [24] P. Kamousi, T. M. Chan, and S. Suri. Closest pair and the post office problem for stochastic points. *Computational Geometry*, 47(2):214–223, 2014.
- [25] D. R. Karger and M. Minkoff. Building steiner trees with incomplete global knowledge. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 613–623, 2000.
- [26] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison Wesley, 2005.
- [27] S. Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222:45–58, 2013.
- [28] S. Li. On facility location with general lower bounds. *CoRR*, abs/1805.02244, 2018.
- [29] M. Mahdian, Y. Ye, and J. Zhang. A 2-approximation algorithm for the soft-capacitated facility location problem. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, pages 129–140, 2003.

- [30] N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1):182–196, 1984.
- [31] S. Nakano. A simple algorithm for r -gatherings on the line. In *Proceedings of WALCOM: Algorithms and Computation*, volume 10755 of Lecture Notes in Computer Science, pages 1–7, Cham, 2018. Springer International Publishing.
- [32] T. Nishizeki and M. S. Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004.
- [33] M. S. Rahman. *Basic Graph Theory*. Undergraduate Topics in Computer Science. Springer, 2017.
- [34] S. M. Ross. *Introduction to Probability Models*. Elsevier, 2014.
- [35] A. Sarker, W. Sung, and M. S. Rahman. A linear time algorithm for the r -gathering problem on the line (extended abstract). In *Proceedings of WALCOM: Algorithms and Computation*, volume 11355 of Lecture Notes in Computer Science, pages 56–66, Cham, 2019. Springer Nature Switzerland.
- [36] K. G. Shin, X. Ju, Z. Chen, and X. Hu. Privacy protection for users of location-based services. *IEEE Wireless Communications*, 19(1):30–39, 2012.
- [37] D. B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, USA*, pages 265–274, 1997.
- [38] L. V. Snyder. Facility location under uncertainty: a review. *IIE Transactions*, 38(7):547–564, 2006.
- [39] S. Suri and K. Verbeek. On the most likely voronoi diagram and nearest neighbor searching. *International Journal of Computer Geometry Applications*, 26(3-4):151–166, 2016.
- [40] Z. Svitkina. Lower-bounded facility location. *ACM Transactions on Algorithms*, 6(4):69:1–69:16, 2010.

- [41] L. Sweeney. k -anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
- [42] Y. Tao, X. Xiao, and R. Cheng. Range search on multidimensional uncertain data. *ACM Transaction on Database Systems*, 32(3):15, 2007.
- [43] H. Wang and J. Zhang. One-dimensional k -center on uncertain data. *Theoretical Computer Science*, 602:114–124, 2015.
- [44] D. B. West. *Introduction to Graph Theory*. Prentice Hall, 1996.
- [45] M. L. Yiu, N. Mamoulis, X. Dai, Y. Tao, and M. Vaitis. Efficient evaluation of probabilistic advanced spatial queries on existentially uncertain data. *IEEE Transaction on Knowledge and Data Engineering*, 21(1):108–122, 2009.
- [46] J. Zeng, G. Telang, M. P. Johnson, R. Sarkar, J. Gao, E. M. Arkin, and J. S. B. Mitchell. Mobile r -gather: Distributed and geographic clustering for location anonymity. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Mobihoc '17, pages 7:1–7:10. ACM, 2017.

Index

adjacent, 10
approximation algorithm, 17
asymptotic behavior, 14
complete graph, 12
complexity, 13
Cumulative Distribution Function, 19
cycle, 12
deterministic algorithm, 15
directed graph, 11
edge, 10
end-vertices, 11, 12
Expected value, 19
graph, 10
Histogram, 19
incident, 11
internal node, 13
leaf, 13
linear-time algorithm, 15
loop, 11
multigraph, 11
multiple edges, 11
neighbor, 11
nondeterministic algorithm, 15
NP-hard, 16
polynomial-time algorithm, 15
Probability Density Function, 18
random variable, 18
root, 13
rooted tree, 12
running time, 14
simple graph, 11
sub-path, 12
tree, 12
undirected graph, 11
Uniform Random Variable, 19
vertex, 10
walk, 12