

Algorithms for Clustering, Partitioning, and Planning

by

Erin C. Taylor

Department of Computer Science
Duke University

Date: _____

Approved:

Pankaj K. Agarwal, Supervisor

Rong Ge

Kamesh Munagala

Jun Yang

Dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Computer Science
in the Graduate School of
Duke University

2023

ABSTRACT

Algorithms for Clustering, Partitioning, and Planning

by

Erin C. Taylor

Department of Computer Science
Duke University

Date: _____

Approved: _____

Pankaj K. Agarwal, Supervisor

Rong Ge

Kamesh Munagala

Jun Yang

An abstract of a dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Computer Science
in the Graduate School of
Duke University

2023

Copyright © 2023 by Erin C. Taylor
All rights reserved

Abstract

We explore three geometric optimization problems with multifaceted goals, including efficiency, fairness, solution quality, and interpretability. The problems we study pose significant challenges because they involve complex geometric objects that interact in high-dimensional spaces. To contend with these challenges, our work utilizes realistic input structure to design algorithms that provide provably good solutions. Alternatively, we can approach the general problem statement with heuristic or approximate schemes.

The first problem domain addressed is trajectory clustering, where the goal is to partition noisy trajectories (polygonal curves) into clusters based on shared structure and find a representative trajectory for each cluster. We present a near-linear time $(1 + \varepsilon)$ -approximation algorithm for k -median clustering of polygonal trajectories under the discrete Fréchet distance and finite point sets under the Hausdorff distance. The algorithm leverages the geometric properties of the trajectories, specifically that centers come from a “simpler” metric space, to achieve efficient clustering results.

The second problem we consider is redistricting, aiming to partition a given population into geographic regions in a fair and representative way. Redistricting involves drawing boundaries for electoral districts, which can significantly impact political representation and decision-making processes. We introduce a new model of fairness for the problem of fair redistricting and investigate the existence and computability of fair partitions. We consider this problem in both one and two dimensions. We also propose a new method of generating redistricting plans that leads to more compact (geometrically “nice”) districts.

The final problem addressed is optimal motion planning for multiple robots in polygonal environments with obstacles. Each robot is represented as a simple poly-

onal object, such as a unit disc or square. We present algorithms for generating motion plans which approximately minimize the sum of distances traveled from a start to target configuration. For the simplified case of two robots and given the paths the robots must traverse, we also give an algorithm to minimize the makespan (latest arrival time) of our schedule.

Contents

Abstract	iv
List of Tables	x
List of Figures	xi
Acknowledgements	xiii
1 Introduction	1
1.1 Trajectory Clustering	2
1.2 Redistricting	3
1.3 Optimal Motion Planning	5
2 k-Median Clustering under discrete Fréchet and Hausdorff distances	8
2.1 Introduction	8
2.2 Preliminaries	14
2.3 Clustering via sampling	18
2.3.1 Correctness of CLUSTER	18
2.3.2 Running time of CLUSTER	22
2.4 Covering metric spaces	25
2.4.1 Sufficient conditions for the strong sampling property	25
2.4.2 Sufficient conditions for the weak sampling property	28
2.5 Clustering discrete Fréchet and Hausdorff distances	34
2.6 Hardness and unbounded doubling dimension	37
2.6.1 Hardness of clustering under the Hausdorff distance	38
2.6.2 Doubling dimension of discrete Fréchet and Hausdorff distances	38

2.7	Conclusion	39
3	Locally Fair Partitioning	41
3.1	Introduction	41
3.1.1	Our Results	42
3.1.2	Related Work	45
3.2	Preliminaries	47
3.3	Existence of Locally Fair Partitions	50
3.4	Clustered Instances	55
3.5	Partitioning Algorithm	59
3.6	Conclusion	64
4	Redistricting via Local Fairness	66
4.1	Introduction	66
4.1.1	Related Work	68
4.1.2	Our Contribution	70
4.2	Model and Preliminaries	71
4.3	Hardness	74
4.4	Heuristics for Efficiently Auditing Local Fairness	77
4.4.1	Ensemble-based Auditing	77
4.4.2	Auditing via Dynamic Programming on Trees	78
4.4.3	Speeding up the DP and Sufficiency of Ensemble-based Auditing	81
4.5	Experiments	84
4.5.1	Datasets and Methods	84
4.5.2	Locally Fair Plans: Counts and Visualizations	87
4.5.3	Compatibility with Extant Fairness and Compactness Notions	89

4.5.4	Actual Plans	90
4.5.5	Sufficiency of Ensemble-based Auditing	90
4.6	Alternative Fairness and Compactness Metrics	93
4.7	Robustness of Local Fairness to Voting Patterns	95
4.8	Visualization of Fair and Unfair Plans for Other States	96
4.9	Conclusions	97
5	Redistricting via Random Geometric Partitioning	99
5.1	Introduction	99
5.1.1	Our Contribution	101
5.1.2	Related Work	102
5.2	Preliminaries	104
5.3	A New Family of Markov Chains	106
5.3.1	Geometric ReCom Proposal	107
5.3.2	Behavior of Markov Chain on Simple Graphs	109
5.4	Experiments	111
5.4.1	Datasets and Methods	112
5.4.2	Sampling Space	114
5.5	Conclusion	115
6	Multi-Robot Motion Planning for Unit Discs with Revolving Areas	117
6.1	Introduction	117
6.2	Hardness of Optimal MRMP-RA	123
6.3	Algorithm	129
6.3.1	Modifying initial path	132
6.3.2	Retracting a robot	132

6.4	Correctness and Analysis of the Algorithm	134
6.4.1	Feasibility	135
6.4.2	Cost of path ensemble	138
6.4.3	Running-time Analysis	143
6.5	Computing a Good Ordering	144
6.6	Makespan	146
6.7	Conclusion	149
7	Conclusion	150
	Bibliography	152
	Biography	163

List of Tables

2.1	Results for (k, C) -median clustering.	8
4.1	Properties of precinct graph data.	85
4.2	Percent of Locally Fair Plans.	86

List of Figures

2.1	A single cluster of trajectories.	10
2.2	(k, C) -median clustering algorithm.	17
2.3	$(1, C)$ -median algorithm.	29
2.4	Recursive Sampling Algorithm.	31
3.1	Example of a Locally Fair Partition.	49
3.2	Instance with no locally fair solution.	54
3.3	Partitioning subproblem.	57
3.4	Dynamic Programming Algorithm to compute Locally Fair Partition.	59
4.1	Example of Fair vs. Unfair plans.	86
4.2	Geographic Distribution of Deviating Groups	87
4.3	Metrics of interest on Ensemble.	88
4.4	Strength of Deviating Groups found by Dynamic Program.	92
4.5	Deviating Groups found by Dynamic Programming Algorithm.	92
4.6	Distribution of Compactness and Partisanship.	94
4.7	Comparison of Election years.	95
4.8	Fair Plan Visualization in MD	96
4.9	Fair Plan Visualization in MI	97
5.1	Ellipse Fitting.	108
5.2	Geometric ReCom Algorithm	109

5.3	Algorithm Comparison.	110
5.4	Grid Graph Example	111
5.5	Grid Graph Comparison.	112
5.6	Compactness Scores.	114
5.7	Maintaining other Metrics.	114
5.8	Percentile Rank Comparison.	115
6.1	Example motion planning instance.	120
6.2	MRMP-RA Hardness Construction.	124
6.3	Revolving Area.	130
6.4	Edited path for two discs.	131
6.5	Sector Type Retraction.	133
6.6	Intersection Type Retraction.	134
6.7	Properties of Revolving Areas.	135
6.8	Segments in Revolving Area.	136
6.9	Distance between retraction points.	137
6.10	Cost of Sector Retraction.	140
6.11	Cost of Intersection Retraction.	140

Acknowledgements

I would like to express my deepest gratitude to my advisor, Pankaj K. Agarwal, for his unwavering guidance, insights, and support throughout the entirety of my research journey. His expertise, encouragement, and patience have been instrumental in shaping the direction and quality of this work from the very beginning of my interest in algorithms.

I would also like to thank Kamesh Munagala, whom I have worked closely with throughout my entire research career, and whom I consider my co-advisor in all but formality. I am also indebted to the other thesis committee members, Rong Ge and Jun Yang, for their invaluable feedback, questions, and exemplars of excellent scholarship.

I would like to extend my appreciation to my collaborators, including Abhinandan Nath, Shao-Heng Ko, Zeyu Shen, Alex Steiger, Dan Halperin, and Tzvika Geft. I am grateful for their time and effort, as this dissertation would not have been achievable without it. I am also thankful to other individuals who generously shared their knowledge and provided helpful discussions, including Hsien-Chih Chang, Brandon Fain, and Chris Tralie.

Thanks to my fellow students, Aaron, Alex, Allen, Govind, Keegan, Rahul, and Stavros, for making my time as a graduate Duke enjoyable. I would also like to thank my teachers and mentors before graduate school for their generosity and support.

Finally, I want to express my heartfelt gratitude to my family and friends for their understanding and encouragement throughout this journey. While the list of individuals mentioned here is not exhaustive, I am sincerely thankful to all those who have played a part in helping me complete this dissertation. Their contributions have made this achievement possible.

Chapter 1

Introduction

The convergence of the physical and virtual realms through modern technology has led to an unprecedented generation of vast amounts of data. Although this data is often high-dimensional and complex, much of it has geometric structure that can be leveraged by algorithms. These algorithms enable researchers to develop insights and solve problems in various fields, including data science, machine learning, GIS systems, and computer graphics.

Due, in part, to the exponential growth of available data, algorithmic problem-solving has become an integral part of various domains, ranging from computer science to finance, healthcare to transportation. These problems often involve complex optimization tasks that demand efficient and effective solutions. However, the very nature of these problems presents significant challenges, even in their most basic settings. Many questions also encompass diverse goals, such as efficiency, fairness, solution quality, and explainability, each of which demands careful consideration.

Nevertheless, by incorporating domain-specific knowledge and constraints, we can tailor algorithms that are theoretically sound and practically applicable. This dissertation demonstrates how we can use geometric techniques to design efficient algorithms that analyze complex data and solve a variety of high-dimensional problems. By leveraging the structure present in realistic input data, our approaches aim to present simple algorithms that perform well in the face of difficult optimization questions. Our techniques exploit geometric properties and relationships within the data to guide algorithms toward accurate and effective solutions. Thus, we endeavor to design algorithms that account for biases, address potential inequalities, and provide provable and interpretable solutions. Toward this end, we focus on three different problem domains: clustering trajectories, redistricting, and optimal motion planning.

1.1 Trajectory Clustering

Trajectories model a variety of systems that change over time, such as data from GPS traces. Since individual trajectories contain noise and uncertainty, partitioning the trajectories into clusters, or groups of trajectories, allows us to better understand the data. Our goal is for the trajectories within each cluster to resemble each other; furthermore, for each cluster, we want to pick a trajectory that represents those in the cluster. This process can be viewed as a compression scheme that performs non-linear dimension reduction. These representative trajectories are useful in a variety of contexts (e.g., route planning, similarity search, anomaly detection), as they are less prone to the noise found in individual trajectories. Clustering trajectories is often a first step to summarizing and compressing GPS data while preserving high-level shared structure.

We focus on studying the k -median clustering problem, a well-studied formulation of the general clustering problem. The main objective is to choose k “centers” in such a way that the total distance from each input element to its nearest cluster center is minimized. In a basic scenario, the elements belong to \mathbb{R}^d and the distances are calculated using the Euclidean metric. In our work, each element itself consists of multiple points, like a trajectory or a point cloud in \mathbb{R}^d . Our research focuses on the discrete Fréchet and Hausdorff distances, which are commonly used for trajectories and point sets, respectively.

The k -median problem is hard to solve exactly, even in Euclidean space. Previous work has proposed approximations algorithms, whose efficiency depends on k and the dimension of the metric space. However, many of these approaches require the metric space to have a bounded doubling dimension. Unfortunately, we prove that neither the discrete Fréchet nor Hausdorff distances satisfy have doubling dimension bounded by a constant, rendering previous methods ineffective.

On the positive side, we give the first near-linear-time $(1 + \varepsilon)$ -approximation algorithm for k -median clustering of polygonal trajectories under the discrete Fréchet distance, and finite point sets under the Hausdorff distance (provided the complexity of each cluster center, ambient dimension, and k are bounded by a constant). Our approach provides a

general framework for solving clustering problems where the cluster centers are restricted to come from a “simpler” metric space. We precisely characterize conditions on the simpler metric space of the cluster centers that allow faster $(1 + \varepsilon)$ -approximations for the k -median problem. We also show that the k -median problem under Hausdorff distance is NP-HARD. The results on trajectory clustering appear in Chapter 2 and are based on joint work with Abhinandan Nath [NT21].

1.2 Redistricting

One challenge in designing algorithms for large-scale decision-making is eliminating bias, which could result in unfair outcomes for specific population segments. For example, for some problems finding a globally optimal solution may systematically underserve subsets of the population. Geometric concepts are also relevant for modeling attributes of individuals and for defining a suitable model of fairness. For instance, consider the fair representation problem, in which we want to partition a population into groups so that each group has a representative chosen by attributes of individuals within the group. Population groups may be defined as geometric regions with natural desirable geometric properties (e.g., compactness, convexity, balanced sizes). This problem is challenging in that we want to consider the space of all geometric k -partitions, which is not a well-characterized space. In addition to geometric structure, we want to ensure that each segment of the population is satisfied with their representative. In this vein, we define a new model of fairness for the problem of fair redistricting and study whether fair partitions of populations based on some criteria always exist and whether they are computable.

More specifically, given a set of n points in the plane, each belonging to one of two parties, and a parameter k , our goal is to compute a partition Π of the plane into regions such that each region contains roughly $\sigma = n/k$ points and Π satisfies a criterion known as “local” fairness. Each region is associated with the majority party in that region, and a point is *unhappy* in Π if it belongs to the minority party. A group D of roughly σ contiguous

points is called a *deviating* group with respect to Π if majority of points in D are unhappy in Π . Such a set of points have a justified complaint with how the partition was drawn. The partition Π is *locally fair* if there is no deviating group with respect to Π . This idea of local fairness is related to the well-studied notion of the core from cooperative game theory.

We first focus on a restricted case when points lie in 1D. The problem is non-trivial even in this setting. We consider both adversarial and “beyond worst-case” settings for this problem. For the former, we characterize the input parameters for which a locally fair partition always exists; we also show that a locally fair partition may not exist for certain parameters. We then consider input models where there are “runs” of red and blue points. For such clustered inputs, we show that a locally fair partition may not exist for certain values of σ , but an approximate locally fair partition exists if we allow some regions to have smaller sizes. We finally present a polynomial-time algorithm for computing a locally fair partition if one exists. These results appear in Chapter 3 and are based on joint work with Pankaj K. Agarwal, Shao-Heng Ko, and Kamesh Munagala [AKMT22].

In Chapter 4, we extend the concept of *local fairness* for auditing and ranking redistricting plans. For redistricting problems on real data, we typically work on a graph, where each vertex represents an indivisible geographic unit. We show that the problem of auditing a given plan for local fairness is NP-complete. We present an Markov Chain Monte Carlo (MCMC) approach for auditing as well as ranking redistricting plans. We also present a dynamic programming based algorithm for the auditing problem that we use to demonstrate the efficacy of our MCMC approach. Using these tools, we test local fairness on real-world election data, showing that it is indeed possible to find plans that are almost or exactly locally fair. Further, we show that such plans can be generated while sacrificing very little in terms of compactness and existing fairness measures such as competitiveness of the districts or seat shares of the plans. This chapter is based on joint work with Pankaj K. Agarwal, Shao-Heng Ko, and Kamesh Munagala [KTAM22].

To complete our discussion of redistricting, in chapter 5, we focus on the increasingly popular MCMC methods for generating and auditing redistricting plans. These methods

generate a large ensemble of representative plans that satisfy desirable properties, such as competitiveness, compactness, and seat share. An important question that arises is how the chain should be constructed so that it runs efficiently, generates an ensemble of desirable plans in terms of the above outcomes, and covers the space of desirable plans in the sense that the ensemble contains a plan close to any desirable plan (a property we term “coverage”). We propose a novel MCMC method that adapts the widely used recombination chain to incorporate geometric partitioning. Our experimental results show that this method achieves better compactness scores, measured both via graph cuts and geometric properties, compared to previous methods, while preserving their competitiveness and similar seat share outcomes. Though our method generates an ensemble from a smaller space of plans than traditional methods, we show that this ensemble achieves good coverage. Finally, our method is computationally efficient and gracefully handles large-scale redistricting problems. This chapter is based on joint work with Pankaj K. Agarwal, Kamesh Munagala, and Zeyu Shen.

1.3 Optimal Motion Planning

The final problem we consider is motion planning. As the fields of artificial intelligence and robotics continue to develop, it has become increasingly essential to develop efficient and reliable decision processes. Moreover, many applications involve the coordination and orchestration of multiple robots, which increases the difficulty of the underlying algorithmic tasks. For example, warehouses for large businesses such as Amazon rely on teams of robots to transport goods in the presence of obstacles. These obstacles also raise the complexity of designing robot motion plans.

More formally, in our motion planning problem, we are given a collection of robots, along with their initial and target positions in some continuous, polygonal environment, which may contain obstacles. Each robot is modeled as a unit disc. Our goal is to find a plan, or set of paths, for the robots that avoids robot-robot and robot-obstacle collisions.

We assume that the robots are labeled, i.e., each robot has a specific target position, and the robots are not interchangeable. We consider two objective functions: minimizing the sum of the distances traveled by the robots, and minimizing the makespan (i.e., latest arrival time) across all robots where their speed is limited to at most unit speed. We model each robot as a unit disc, and we consider polygonal environments. We note that this problem has a large number of degrees of freedom: the placement of each robot can be represented by 2 parameters, so the entire placement of a system of k robots has $2k$ degrees of freedom.

We study the problem of motion planning for a collection of n labeled unit disc robots in a polygonal environment. We assume that the robots have *revolving areas* around their start and final positions: that each start and each final is contained in a radius 2 disc lying in the free space, not necessarily concentric with the start or final position, which is free from other start or final positions. This assumption allows a *weakly-monotone* motion plan, in which robots move according to an ordering as follows: during the turn of a robot R in the ordering, it moves fully from its start to final position, while other robots do not leave their revolving areas. As a moving robot passes through a revolving area, there is enough space inside the revolving area for another robot to maneuver to avoid a collision. Notwithstanding the existence of a motion plan, we show that minimizing the total traveled distance in this setting, specifically even when the motion plan is restricted to be weakly-monotone, is APX-hard, ruling out any polynomial-time $(1 + \varepsilon)$ -approximation algorithm.

For this problem, we present the first constant-factor approximation algorithm for computing a feasible weakly monotone motion plan. (Note that the optimal motion plan need not be weakly monotone.) Moreover, our algorithm extends to an online setting in which the polygonal environment is fixed but the initial and final positions of robots are specified in an online manner. We also consider the problem of finding the optimal ordering of robots to avoid robot-robot collisions. This task is important because, as we show, the overhead in the total cost we incur while editing the paths varies significantly depending on this ordering. It is known that solving this problem optimally is NP-hard; we provide an $O(\log n \log \log n)$ -approximation algorithm. These results are joint work with Pankaj

K. Agarwal, Tzvika Geft, and Dan Halperin [AGHT22].

For the makespan objective, even in the absence of obstacles, characterizing the optimal paths has been a challenging research question. Instead, we show how to schedule movement along given paths for two robots to achieve the optimal makespan among all feasible schedules restricted to the given paths. This result is joint work with Pankaj K. Agarwal and Alex Steiger. All of the results on motion planning appear in Chapter 6.

Chapter 2

k -Median Clustering under discrete Fréchet and Hausdorff distances

2.1 Introduction

We study the k -median problem for an arbitrary metric space $\mathcal{X} = (X, \mathbf{d})$, where the cluster centers are restricted to come from a (possibly infinite) subset $C \subseteq X$. We call it the (k, C) -median problem. We prove general conditions on the structure of C that allow us to get efficient $(1 + \varepsilon)$ -approximation algorithms for the (k, C) -median problem for any $\varepsilon > 0$. As applications of our framework, we give $(1 + \varepsilon)$ -approximation algorithms for the metric space defined over polygonal trajectories and finite point sets in \mathbb{R}^d under the discrete Fréchet and Hausdorff distance respectively, where the cluster centers have bounded complexity. For trajectories, our algorithm runs in near-linear time in the number of input points (Theorem 16) and is significantly faster than the previous best algorithm ([BDS19], Theorem 11). For point sets, ours is the first $(1 + \varepsilon)$ -approximation algorithm that runs in linear time in the number of input points (Theorem 18) for bounded dimensions and cluster complexity. Our results are summarized in Table 2.1. We also show that the k -median problem under Hausdorff distance is NP-HARD.

Table 2.1: Our results for $(1 + \varepsilon)$ -approximate k -median for n input trajectories/point sets in \mathbb{R}^d , each having at most m points. Each cluster center can have at most l points. While stating running times, we assume k, l and d are constants independent of n, m , and \tilde{O} hides logarithmic factors in n, m .

Metric space	Our result	Previous best
Polygonal traj., discrete Fréchet	$\tilde{O}(nm)$	$\tilde{O}(n^{dkl+1}m)$ [BDS19]
Point sets, Hausdorff	$O(nm)$	–

The k -median problem has been widely studied. We are given a set P of n elements from a metric space. The goal is now to select k *centers* so that the sum of the distances of each element to the nearest cluster center is minimized. In the simplest setting, $P \subseteq \mathbb{R}^d$ under the Euclidean metric. In this chapter, each individual element of P is itself a collection of points in \mathbb{R}^d , e.g., a curve traced by a moving object, or a point cloud. Since the objective of clustering is to group *similar* objects into the same cluster and to summarize each cluster using its cluster center, it is important that a meaningful distance function is used to compare two input elements. In our work, we look at the widely used discrete Fréchet and Hausdorff distances for trajectories and point sets respectively.

Trajectories can model a variety of systems that change with time. As such, trajectory data is being collected at enormous scales. As a first step, clustering is hugely important in understanding and summarizing the data. It involves partitioning a set of trajectories into clusters of similar trajectories, and computing a representative trajectory (a center) per cluster. It can be viewed as a compression scheme for large trajectory datasets, effectively performing non-linear dimension reduction. If the centers have low complexity, this representation can reduce uncertainty and noise found in individual trajectories. Information provided by the set of centers is useful for trajectory analysis applications such as similarity search and anomaly detection [SFR12]. The Hausdorff distance is another widely used shape-based distance [BGLR16, HKR93]. In shape matching applications, we may want to cluster similar shapes into one group (where a shape is represented by a point cloud).

The k -median problem is hard to solve exactly, even in Euclidean space [FG88, MS84]. There is a long line of work on both constant factor and $(1 + \varepsilon)$ -approximations, with varying running time dependence on k and the ambient dimension (if applicable). Many of these algorithms require the underlying metric space to have bounded doubling dimension (e.g., see [ABS10]). However, it can be shown that both the discrete Fréchet and Hausdorff distances do not have doubling dimension bounded by a constant (Section 2.6.2). We circumvent this problem by considering cluster centers from a somewhat *simpler* metric space compared to the input metric space. For trajectories and point sets, we restrict

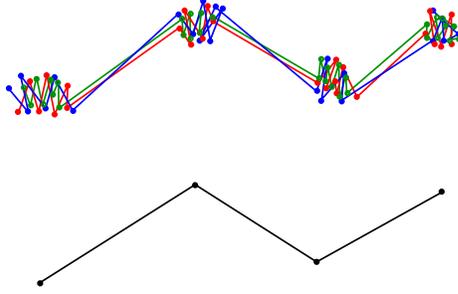


Figure 2.1: The red, green and blue trajectories are similar to each other and form a single cluster. If the cluster center trajectory is restricted to have four vertices, it will look like the black trajectory at the bottom. However if the center is unrestricted, it will contain a lot of vertices inside the four noticeable noisy *clumps* of vertices of the input trajectories, thereby overfitting to the input.

each center to have low complexity, i.e., a bounded number of points. This approach has been used before ([BDG⁺19, BDS19, DKS16]). It has the added benefit of preventing the cluster center from overfitting to the elements of its cluster. This is crucial, since real-life measurements are noisy and error-prone, and without any restrictions the cluster center can inherit noise and high complexity from the input (see Fig. 2.1). As another example, in clustering financial time-series data using Hausdorff distance [BBDC⁺07], frequent intraday fluctuations may not be useful in capturing long-term trends, and we want to avoid them by restricting the cluster centers' complexity. However, our work differs from previous approaches in that we precisely characterize general conditions on the simpler metric space for the cluster centers, which leads to faster $(1 + \varepsilon)$ -approximation algorithms for the k -median problem for the discrete Fréchet and Hausdorff distances.

Problem definition. A *metric space* $\mathcal{X} = (X, d)$ consists of a set X and a distance function $d : X \times X \rightarrow \mathbb{R}_{\geq 0}$ that satisfies the following properties : (i) $d(x, x) = 0$ for all $x \in X$; (ii) $d(x, y) = d(y, x)$ for all $x, y \in X$; and (iii) $d(x, z) \leq d(x, y) + d(y, z)$ for all $x, y, z \in X$.

Given subsets $P, C \subseteq X$, the (k, C) -*median* problem is to compute a set $C' \subseteq C$ of k

center points that minimizes

$$\sum_{p \in P} \mathbf{d}(p, C'),$$

where $\mathbf{d}(p, C') = \min_{c \in C'} \mathbf{d}(p, c)$. Here P is finite, but C need not be.

Let T^l (resp. U^l) be the set of all trajectories (resp. point sets) in \mathbb{R}^d , where each trajectory (resp. point set) has at most l points. Thus, $T = \bigcup_{l>0} T^l$ and $U = \bigcup_{l>0} U^l$ are the set of all trajectories and finite point sets in \mathbb{R}^d respectively. As special cases of the (k, C) -median problem, we discuss the (k, T^l) -median and the (k, U^l) -median problems for the metric spaces $\mathcal{T} = (T, \mathbf{d}_F)$ and $\mathcal{U} = (U, \mathbf{d}_H)$ respectively, i.e., each center trajectory or point set can have at most l points. Here, \mathbf{d}_F and \mathbf{d}_H denote the discrete Fréchet and Hausdorff distances respectively.

1

Challenges and ideas. As mentioned before, both the discrete Fréchet and Hausdorff metrics do not have low doubling dimension, so a number of previous techniques do not directly apply to our setting to yield efficient algorithms. One approach would be to embed these metrics into other metric spaces. Backurs and Sidiropoulos [BS16] give an embedding of the Hausdorff metric over point sets of size s in d -dimensional Euclidean space, into $l_\infty^{s^{O(s+d)}}$ with distortion $s^{O(s+d)}$. However, both the distortion and the resultant dimension are too high for many applications. It is not known if the Fréchet distance can be embedded into an l_p space using finite dimension.

We circumvent the problem by restricting the cluster centers to come from a subset C of the original space X , namely trajectories and point sets defined by a bounded number of points each. As mentioned earlier, this approach was used before as well for the k -median problem under the discrete Fréchet distance [BDS19], where the input has n polygonal trajectories and at most m points each, and the cluster centers are polygonal trajectories with at most l points. They obtain a $(1 + \varepsilon)$ -algorithm by first running a fast constant

¹Note that the term *points* can refer to both elements of a metric space X , as well as to the points of the individual trajectories/point sets. However, the usage should be clear from the context.

factor approximation algorithm, then enumerating all subsets of k trajectories (with at most l vertices each) whose vertices lie on hypercube grid points around the vertices of the approximation, then choosing the subset with minimum cost. However, the number of such grid points depends on n for $k > 1$, causing a $O(n^{dkl})$ factor in their running time. On the other hand, we describe a more general framework that takes advantage of the *simpler* structure of C (compared to X), and show that if every metric ball in C can be *covered* by a small number of metric balls of a fixed smaller radius, then we can use a sampling-based algorithm similar to the one in Ackermann *et al.* [ABS10]; we make this precise by introducing the notion of *coverability* of C . We crucially show that the centers of the balls in the *cover* can be arbitrary, and need not come from C . This is more general than C having bounded doubling dimension. This allows us to approximate the optimal $(1, C)$ -median of P using the $(1, C)$ -median of a constant sized random sample of P , allowing us to use the framework of [ABS10].

It is not known how to efficiently compute the optimal $(1, C)$ -median under the discrete Fréchet and Hausdorff distances. However, we show that all we need is to compute a constant number of candidate centers in time independent of the size of the input, at least one of which is a good approximation to the optimal $(1, C)$ -median of the input. We show how to compute these candidates for a *coverable* set C . Then, we can apply the sampling technique of [ABS10] and recursively use this property to find k centers that approximate the cost of the (k, C) -median optimal solution. Although our work heavily relies on the framework of Ackermann *et al.*[ABS10], it is a significant improvement from existing work on clustering under the discrete Fréchet distance, and the first such result for clustering under the Hausdorff distance.

Previous work. Trajectory clustering has a lot of practical applications, e.g., discovering frequent movement patterns in trajectory data. As such, there has been work on trajectory clustering [GS99, HPL15, XZLZ15], and possibly computing a representative trajectory for each cluster. Many proposed algorithms and models have no provable performance guarantees and are experimental in nature.

Driemel *et al.* [DKS16] started the rigorous study of clustering trajectories under the continuous Fréchet distance under the classic k -clustering objectives. However, they only deal with 1D-trajectories. They introduce the (k, l) -clustering problem, i.e., clustering trajectories with k cluster centers such that each center can have at most l points. For 1D-trajectories, they give $(1 + \varepsilon)$ -approximation algorithms for both the (k, l) -center and (k, l) -median problem that run in near-linear time for constant ε, k, l . Buchin *et al.* [BDG⁺19] study the (k, l) -center clustering problem for trajectories in \mathbb{R}^d under the discrete and continuous Fréchet distances, and give both upper and lower bounds. The most closely related work to ours is the one by Buchin, Driemel and Struijs [BDS19], where they give algorithms for the (k, l) -median problem under discrete Fréchet distance; however their running times are much slower (see Table 2.1). They also show that the 1-median problem under discrete Fréchet distance is NP-HARD, and W[1]-HARD in the number of input trajectories.

On the other hand, clustering under Hausdorff distance has received much less attention. There is work on hierarchical clustering of financial time series data using Hausdorff distance [BBDC⁺07]. Chen *et al.* [CWLS11] use the DBSCAN algorithm [EK SX96] while Qu *et al.* [QZC09] use spectral clustering for trajectories and using the Hausdorff distance. We are not aware of any theoretical analysis for k -median clustering of point sets under the Hausdorff distance.

In general metric spaces, a polynomial time $(1 + \sqrt{3} + \varepsilon)$ -approximation algorithm to the k -median problem exists [LS16], whereas no polynomial time algorithm can achieve an approximation ratio less than $(1 + 2/e)$ unless $\text{NP} \subseteq \text{DTIME}[n^{O(\log \log n)}]$ [JMS02]. For the Euclidean k -median problem in d dimensions, Guruswami and Indyk [GI03] showed that there is no PTAS for k -median if both k and d are part of the input. Arora *et al.* [ARR98] gave the first PTAS when d is fixed, whose running time was subsequently improved in [KR07]. Kumar *et al.* [KSS10] gave a $(1 + \varepsilon)$ -approximate algorithm with runtime $2^{(k/\varepsilon)^{O(1)}} dn$, this was extended by Ackermann *et al.* [ABS10] to those metric spaces for which the optimal 1-median can be approximated using a constant-sized random sample;

this holds true for doubling metric spaces. There are coresets-based approaches with running times linear in n and either exponential in d and polynomial in k [HPM04, HPK07], or vice versa [BHPI02]. Recently, Cohen-Addad *et al.* [CAKM19] gave a PTAS for k -median in low-dimensional Euclidean and minor-free metrics using local search.

We also note that subsequently after publication, there was a line of followup work giving algorithms for k -median clustering under the continuous Fréchet distance, see [MMR19, BDR23].

2.2 Preliminaries

We formally define the discrete Fréchet and Hausdorff distances. We also define properties on C and d which allow us to design efficient clustering algorithms. Finally we give an overview of our algorithm.

Discrete Fréchet and Hausdorff distance. Consider two finite sets ζ_1 and ζ_2 . A *correspondence* \mathcal{C} between ζ_1 and ζ_2 is a subset of $\zeta_1 \times \zeta_2$ such that every element of ζ_1 and ζ_2 appears in at least one pair in \mathcal{C} . For $\zeta_1, \zeta_2 \subseteq \mathbb{R}^d$, the *Hausdorff distance* [Hau78] is defined as

$$d_H(\zeta_1, \zeta_2) = \min_{\mathcal{C} \in \Xi(\zeta_1, \zeta_2)} \max_{(p, q) \in \mathcal{C}} \|p - q\|,$$

where $\|\cdot\|$ denotes the l_2 norm, and $\Xi(\zeta_1, \zeta_2)$ is the set of all possible correspondences between ζ_1 and ζ_2 .

A trajectory γ is a finite sequence of points $\langle p_1, p_2 \dots \rangle$ in \mathbb{R}^d . For any integer $m > 0$, let $[m]$ denote the set of the first m positive integers. Given two trajectories $\gamma_1 = \langle p_1, \dots, p_{m_1} \rangle$ and $\gamma_2 = \langle q_1, \dots, q_{m_2} \rangle$, a *walk* \mathcal{W} between γ_1 and γ_2 is an ordered sequence of points in $[m_1] \times [m_2]$ with the first and last members of the sequence being $(1, 1)$ and (m_1, m_2) . Such a walk \mathcal{W} is said to be *monotone* if for every consecutive pair of elements $(i_1, j_1), (i_2, j_2)$ in \mathcal{W} , we have $i_2 \in \{i_1, i_1 + 1\}, j_2 \in \{j_1, j_1 + 1\}$. The *discrete Fréchet distance* [EM94]

between γ_1 and γ_2 is then defined as

$$\mathbf{d}_F(\gamma_1, \gamma_2) = \min_{W \in \Xi_M(\gamma_1, \gamma_2)} \max_{(i,j) \in W} \|p_i - q_j\|,$$

where $\Xi_M(\gamma_1, \gamma_2)$ is the set of all monotone walks between γ_1 and γ_2 . Our algorithms compute a clustering in the metric space defined by the discrete Fréchet and Hausdorff distances.

Strong and weak sampling properties. We define two properties that make efficient clustering algorithms possible. These are generalizations of the *strong* and *weak* sampling properties defined by Ackermann *et al.* (see Theorem 1.1 and Property 4.1 in [ABS10]). The major difference is that the cluster centers are restricted to a subset C of the metric space X . These properties allow fast approximation of the $(1, C)$ -median using only a constant sized random sample of the input. We later show how to get an efficient (k, C) -median algorithm using the fast $(1, C)$ -median algorithm as a subroutine. We denote the optimal $(1, C)$ median of any set $P \subseteq X$ by c_P .

Definition 1 Strong sampling property. *Let $0 < \varepsilon, \delta < 1$ be arbitrary. (X, C, \mathbf{d}) is said to satisfy the strong sampling property for ε, δ iff*

- (i) *For any finite $P \subseteq X$, c_P can be computed in time depending only on $|P|$.*
- (ii) *There exists a positive integer $m_{\delta, \varepsilon}$ depending on δ, ε such that for any $P \subseteq X$, the optimal $(1, C)$ -median c_S of a uniform random multiset $S \subseteq P$ of size $m_{\delta, \varepsilon}$ satisfies*

$$\Pr \left[\sum_{p \in P} \mathbf{d}(p, c_S) \leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P) \right] \geq 1 - \delta.$$

The strong sampling property characterizes those instances in which the optimal $(1, C)$ -median of a constant-sized random sample is a good approximation to the optimal $(1, C)$ -median of the whole set. However, in many cases it is impossible to efficiently solve the $(1, C)$ -median exactly (e.g., when $C, X = \mathbb{R}^d$ and \mathbf{d} is the Euclidean metric). This is also true for the discrete Fréchet and Hausdorff distances for polygonal trajectories and finite point sets respectively. The following definition becomes helpful then.

Definition 2 Weak sampling property. Let $0 < \varepsilon, \delta < 1$ be arbitrary. (X, C, \mathbf{d}) is said to satisfy the weak sampling property for ε, δ iff there exist positive integers $m_{\delta, \varepsilon}$ and $t_{\delta, \varepsilon}$ depending on δ, ε such that for any $P \subseteq X$ and a uniform random multiset $S \subseteq P$ of size $m_{\delta, \varepsilon}$, there exists a set $\Gamma(S) \subseteq C$ of size $t_{\delta, \varepsilon}$ that satisfies

$$\Pr \left[\exists c \in \Gamma(S) \mid \sum_{p \in P} \mathbf{d}(p, c) \leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P) \right] \geq 1 - \delta.$$

Furthermore, $\Gamma(S)$ can be computed in time depending on $\delta, \varepsilon, |S|$ but independent of $|P|$.

The weak sampling property characterizes those instances in which it is possible to generate a constant number of candidate cluster centers in time independent of the size of the input set, and at least one of which is guaranteed to be a good approximation to the optimal $(1, C)$ -median of the input set. We later show that the discrete Fréchet and Hausdorff distances satisfy the weak sampling property. We use $m_{\delta, \varepsilon}$ to denote the size of the random sample for both the strong and weak sampling properties.

Algorithm overview. We give an overview of our algorithm, denoted CLUSTER, in Fig. 2.2. It is similar to the algorithm CLUSTER from [ABS10], with the small but crucial difference being that the set of candidate cluster centers \overline{C}_S comes from C ; this also changes how the candidates are generated. We show that with a careful choice of C , our instance satisfies one of the sampling properties (Definitions 1, 2), and we can apply the framework of Ackermann *et al.*[ABS10].

The algorithm takes as input the set of points $\overline{P} \subseteq P$ that are yet to be assigned cluster centers, the number of cluster centers \overline{k} still to be computed, and the centers $\overline{C} \subseteq C$ already computed. It returns the final set of cluster centers. To solve the (k, C) -median problem for P , we call CLUSTER($P, k, \{\}$) with values of $\alpha, m_{\delta, \varepsilon}, F$ that we will specify later.

Briefly, the algorithm has two phases. In the pruning phase no new centers are added. Rather, the set N containing half of the points of \overline{P} closest to \overline{C} are removed from \overline{P} , and the algorithm is called recursively on $\overline{P} \setminus N$. In the sampling phase, new centers are added. The algorithm first samples a uniformly random multiset S of \overline{P} of size $\frac{2}{\alpha} m_{\delta, \varepsilon}$ for some

```

CLUSTER( $\bar{P}, \bar{k}, \bar{C}$ ):
  input: Point set  $\bar{P}$ , remaining number of centers  $\bar{k}$ , computed centers  $\bar{C}$ 
  if  $\bar{k} = 0$ : return  $\bar{C}$ 
  else:
    if  $\bar{k} \geq |\bar{P}|$ : return  $\bar{C} \cup \{c_p \mid p \in \bar{P}\}$ , where  $c_p$  is such that  $d(p, c_p) \leq (1 + \varepsilon)d(p, C)$ 
    else:
      /* Pruning phase */
       $N \leftarrow$  set of  $\frac{1}{2}|\bar{P}|$  minimal points  $p \in \bar{P}$  w.r.t.  $d(p, \bar{C})$ 
       $C^* \leftarrow$  CLUSTER( $\bar{P} \setminus N, \bar{k}, \bar{C}$ )
      /* Sampling phase */
       $S \leftarrow$  uniform random multisubset of  $\bar{P}$  of size  $\frac{2}{\alpha}m_{\delta, \varepsilon}$ 
       $\bar{C}_S \leftarrow \bigcup_{S' \subset S, |S'|=m_{\delta, \varepsilon}} F(S')$ 
      for all  $\bar{c} \in \bar{C}_S$  :
         $C^{\bar{c}} \leftarrow$  CLUSTER( $\bar{P}, \bar{k} - 1, \bar{C} \cup \{\bar{c}\}$ )
      return  $C^{\bar{c}}$  or  $C^*$  with the lowest cost described in 2.2.

```

Figure 2.2: Algorithm CLUSTER computes a (k, C) -median clustering.

constant α and $m_{\delta, \varepsilon}$ to be defined later. Then for each subset $S' \subset S$ of size $m_{\delta, \varepsilon}$, a set of candidate centers $F(S')$ is generated; the function F varies depending on certain conditions satisfied by (X, C, d) ; in particular $F(S') = \{c_{S'}\}$ for the strong sampling property, and $F(S') = \{\Gamma(S')\}$ for the weak sampling property. Each candidate center is in turn added to \bar{C} , and the algorithm is run recursively. Finally, the solution with the lowest cost is returned.

Organization of the chapter. The rest of the chapter is organized as follows. In Section 2.3, we show that for (X, C, d) satisfying the strong and weak sampling properties, the algorithm CLUSTER (using the appropriate $\alpha, m_{\delta, \varepsilon}, F$) computes a $(1 + \varepsilon)$ -approximation to the optimal (k, C) -median; we also bound the running time. In Section 2.4, we prove sufficient conditions on C for the sampling properties to hold. In Section 2.5, we give clustering algorithms for the discrete Fréchet and Hausdorff distances using the framework developed in previous sections. In Section 2.6, we show that the k -median problem for Hausdorff distance is NP-HARD, and that the doubling dimension for the discrete Fréchet and Hausdorff distances is not bounded by a constant.

2.3 Clustering via sampling

We show that the CLUSTER algorithm (Fig. 2.2) computes an approximate (k, C) -median for instances satisfying the sampling properties and for appropriate $\alpha, m_{\delta, \varepsilon}$ and F , i.e., if we use $F(S') = \{c_{S'}\}$ for the strong sampling property and $F(S') = \Gamma(S')$ for the weak sampling property. The analysis closely follows that of Ackermann *et al.* [ABS10].

2.3.1 Correctness of CLUSTER

The *superset sampling lemma* (Lemma 3) shows how to draw a uniform random multiset from $P' \subseteq P$ while only knowing P (without explicitly knowing P'), provided P' contains a sufficient fraction of the points of P . In our algorithm in Fig. 2.2, we use this property in the *sampling phase* in order to sample points from a single cluster, without explicitly knowing the points in the said cluster.

Lemma 3 *Superset sampling.* Suppose $0 < \alpha < \frac{1}{4}$ and $\varepsilon, \delta \in (0, 1)$. Suppose (X, C, \mathbf{d}) satisfies either the strong or weak sampling property. Let $P \subseteq X$ of size n and $P' \subseteq P$ with $|P'| \geq \alpha n$. Let $S \subseteq P$ be a uniform sample multiset of size at least $\frac{2}{\alpha} m_{\delta, \varepsilon}$. Then with probability at least $\frac{1-\delta}{5}$, there exists a subset $S' \subseteq S$ with $|S'| = m_{\delta, \varepsilon}$ satisfying the following.

$$\exists c \in F(S') \text{ s.t. } \sum_{p \in P'} \mathbf{d}(p, c) \leq (1 + \varepsilon) \sum_{p \in P'} \mathbf{d}(p, c_{P'}).$$

Here, $F(S')$ is either $\{c_{S'}\}$ or $\Gamma(S')$ for (X, C, \mathbf{d}) satisfying the strong or weak sampling property respectively.

Proof. This proof is similar to that of Lemma 2.1 in [ABS10]. Define Y to be a random variable denoting the number of points from P' contained in S . Note that $\mathbf{E}[Y] \geq 2m_{\delta, \varepsilon}$, since S is sampled with replacement. By applying a Chernoff bound, we obtain

$$\Pr[Y < m_{\delta, \varepsilon}] \leq \Pr\left[Y < \frac{\mathbf{E}[Y]}{2}\right] \leq e^{-m_{\delta, \varepsilon}/4} \leq e^{-1/4} < \frac{4}{5},$$

since $m_{\delta, \varepsilon} \geq 1$. Thus, with probability at least $\frac{1}{5}$, S contains at least $m_{\delta, \varepsilon}$ points of P' . This along with the strong or weak sampling property finishes the proof. \square

Using the superset sampling lemma and the strong and weak sampling properties, we now show that the CLUSTER algorithm indeed computes a solution with a good approximation factor.

Lemma 4 Correctness of CLUSTER. *Let $\alpha < \frac{1}{4k}$ be an arbitrary positive constant. Suppose (X, C, \mathbf{d}) satisfies the strong or weak sampling property (Definitions 1, 2) for some $\varepsilon, \delta \in (0, 1)$. Given $P \subseteq X$, algorithm CLUSTER run with input $(P, k, \{\})$ and appropriate F computes a set $\tilde{C} \subseteq C$ of size k such that*

$$\Pr \left[\sum_{p \in P} \mathbf{d}(p, \tilde{C}) \leq (1 + 8\alpha k^2)(1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, C^*) \right] \geq \left(\frac{1 - \delta}{5} \right)^k,$$

where C^* is an optimal solution to the (k, C) -median problem for P .

Proof. We prove the lemma for $k = 2$; it generalizes in a straightforward manner for $k > 2$. We assume $n = |P|$ is a power of 2 for simplicity. Suppose P_1, P_2 are the clusters with centers $C^* = \{c_1^*, c_2^*\}$ corresponding to the optimal $(2, C)$ -median for P . Assume $|P_1| \geq \frac{1}{2}|P_2|$ w.l.o.g. Let $opt_1(Q)$ and $opt_2(Q)$ denote the values of the optimal $(1, C)$ -median and $(2, C)$ -median respectively of any set Q .

By Lemma 3, during the sampling phase of the initial call to CLUSTER, $\overline{C_S}$ contains a $\overline{c_1}$ such that $\sum_{p \in P_1} \mathbf{d}(p, \overline{c_1}) \leq (1 + \varepsilon)opt_1(P_1)$ with probability at least $\frac{1-\delta}{5}$. We consider two cases: when the algorithm selects $\overline{c_1}$ and recurses with $(\overline{P}, 1, \{\overline{c_1}\})$ where \overline{P} contains a suitably large number of points from P_2 , and when this does not occur.

Case 1: Suppose there exists a call with $(\overline{P}, 1, \{\overline{c_1}\})$ such that $|P_2 \cap \overline{P}| \geq \alpha |\overline{P}|$. Then, by Lemma 3, during this call $\overline{C_S}$ contains a $\overline{c_2}$ such that $\sum_{p \in P_2 \cap \overline{P}} \mathbf{d}(p, \overline{c_2}) \leq (1 + \varepsilon)opt_1(P_2 \cap \overline{P})$. In this case, we upper bound the cost of $\overline{C} = \{\overline{c_1}, \overline{c_2}\}$. Let $N = P \setminus \overline{P}$ be the points removed by the pruning phases between the sampling of $\overline{c_1}$ and $\overline{c_2}$. $P_1, P_2 \cap N$, and $P_2 \cap \overline{P}$ form a partition of P . We have that:

$$\sum_{p \in P} \mathbf{d}(p, \overline{C}) \leq \sum_{p \in P_1} \mathbf{d}(p, \overline{c_1}) + \sum_{p \in P_2 \cap N} \mathbf{d}(p, \overline{c_1}) + \sum_{p \in P_2 \cap \overline{P}} \mathbf{d}(p, \overline{c_2}). \quad (2.1)$$

By the definition of $\overline{c_1}$, we can bound the first term of Equation 2.1:

$$\sum_{p \in P_1} \mathbf{d}(p, \bar{c}_1) \leq (1 + \varepsilon) \sum_{p \in P_1} \mathbf{d}(p, c_1^*).$$

Next, we bound the last term of Equation 2.1. By the selection of \bar{c}_2 :

$$\sum_{p \in P_2 \cap \bar{P}} \mathbf{d}(p, \bar{c}_2) \leq (1 + \varepsilon) \sum_{p \in P_2 \cap \bar{P}} \mathbf{d}(p, c_{P_2 \cap \bar{P}}) \leq (1 + \varepsilon) \sum_{p \in P_2} \mathbf{d}(p, c_2^*).$$

Now, we only need to bound the middle term of Equation 2.1. We first assume $N \neq \emptyset$, otherwise we are done. Suppose there are t recursive calls (and thus pruning phases) between sampling \bar{c}_1 and \bar{c}_2 . Then, $N = N^{(1)} \cup \dots \cup N^{(t)}$ where $|N^{(i)}| = \frac{n}{2^i}$, since in each pruning phase $\frac{1}{2}|\bar{P}|$ points are removed. Intuitively, each $N^{(i)}$ contains only a few points from P_2 . Let $\bar{P}^{(0)} = P$ and $\bar{P}^{(i)} = \bar{P}^{(i-1)} \setminus N^{(i)}$. Recall that when $|P_2 \cap \bar{P}| \geq \alpha|\bar{P}|$, we assign these points to \bar{c}_2 . Thus, $|P_2 \cap \bar{P}^{(i)}| < \alpha|\bar{P}^{(i)}|$ for all $i < t$. Thus, we bound the number of points of P_2 in each $N^{(i)}$ for $i \leq t$:

$$|P_2 \cap N^{(i)}| \leq |P_2 \cap \bar{P}^{(i-1)}| < \alpha|\bar{P}^{(i-1)}| = 2\alpha \frac{n}{2^i}, \quad (2.2)$$

because $N^{(i)} \subset \bar{P}^{(i-1)}$. We can also bound the number of points of P_1 in each $N^{(i)}$:

$$|P_1 \cap N^{(i)}| \geq |N^{(i)}| - |P_2 \cap N^{(i)}| \geq (1 - 2\alpha) \frac{n}{2^i}. \quad (2.3)$$

We first show that assigning $P_2 \cap N$ to \bar{c}_1 has small cost. If $p \in N^{(i)}$ and $p' \in N^{(i+1)}$, it must be that $\mathbf{d}(p, \bar{c}_1) \leq \mathbf{d}(p', \bar{c}_1)$ since minimal points with respect to \bar{C} are chosen at each step to be removed. Thus we can sum over such p, p' , and for all $i < t$

$$\begin{aligned} \frac{1}{|P_2 \cap N^{(i)}|} \sum_{p \in P_2 \cap N^{(i)}} \mathbf{d}(p, \bar{c}_1) &\leq \frac{1}{|P_1 \cap N^{(i+1)}|} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1) \\ \Rightarrow \sum_{p \in P_2 \cap N^{(i)}} \mathbf{d}(p, \bar{c}_1) &\leq \frac{|P_2 \cap N^{(i)}|}{|P_1 \cap N^{(i+1)}|} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1). \end{aligned}$$

Combining this with Equations 2.2 and 2.3, for all $i < t$ we obtain

$$\sum_{p \in P_2 \cap N^{(i)}} \mathbf{d}(p, \bar{c}_1) \leq \frac{2\alpha n/2^i}{(1-2\alpha)n/2^{i+1}} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1) \leq \frac{4\alpha}{1-2\alpha} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1). \quad (2.4)$$

Finally, we bound the cost of assigning points in $P_2 \cap N^{(t)}$ to \bar{c}_1 . Our previous observation that minimal points with respect to \bar{C} are pruned at each step yields:

$$\begin{aligned} \frac{1}{|P_2 \cap N^{(t)}|} \sum_{p \in P_2 \cap N^{(t)}} \mathbf{d}(p, \bar{c}_1) &\leq \frac{1}{|P_1 \cap \bar{P}^{(t)}|} \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1) \\ \Rightarrow \sum_{p \in P_2 \cap N^{(t)}} \mathbf{d}(p, \bar{c}_1) &\leq \frac{|P_2 \cap N^{(t)}|}{|P_1 \cap \bar{P}^{(t)}|} \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1). \end{aligned}$$

We further have $|P_1 \cap \bar{P}^{(t)}| = |\bar{P}^{(t)}| - |P_2 \cap \bar{P}^{(t)}| \geq |\bar{P}^{(t)}| - |P_2 \cap \bar{P}^{(t-1)}| > (1-2\alpha)\frac{n}{2^t}$.

Combined with Equation 2.2 we get

$$\sum_{p \in P_2 \cap N^{(t)}} \mathbf{d}(p, \bar{c}_1) \leq \frac{2\alpha n/2^t}{(1-2\alpha)n/2^t} \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1) \leq \frac{2\alpha}{1-2\alpha} \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1). \quad (2.5)$$

Now, we have computed a bound for assigning points of P_2 to \bar{c}_1 in each $N^{(i)}$. Combining these:

$$\begin{aligned} \sum_{p \in P_2 \cap N} \mathbf{d}(p, \bar{c}_1) &= \sum_{i=1}^t \sum_{p \in P_2 \cap N^{(i)}} \mathbf{d}(p, \bar{c}_1) \\ &\leq \frac{4\alpha}{1-2\alpha} \sum_{i=1}^{t-1} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1) + \frac{2\alpha}{1-2\alpha} \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1) \\ &\leq 8\alpha \sum_{i=1}^{t-1} \sum_{p \in P_1 \cap N^{(i+1)}} \mathbf{d}(p, \bar{c}_1) + 8\alpha \sum_{p \in P_1 \cap \bar{P}^{(t)}} \mathbf{d}(p, \bar{c}_1) \\ &\leq 8\alpha \sum_{p \in P_1} \mathbf{d}(p, \bar{c}_1) \\ &\leq 8\alpha(1+\varepsilon) \sum_{p \in P_1} \mathbf{d}(p, c_1^*), \end{aligned}$$

for $\alpha \leq \frac{1}{4}$.

Thus for Case 1, we can bound the algorithm cost from Equation 2.1:

$$\begin{aligned}
\sum_{p \in P} \mathbf{d}(p, \bar{C}) &\leq \sum_{p \in P_1} \mathbf{d}(p, \bar{c}_1) + \sum_{p \in P_2 \cap N} \mathbf{d}(p, \bar{c}_1) + \sum_{p \in P_2 \cap \bar{P}} \mathbf{d}(p, \bar{c}_2) \\
&\leq (1 + \varepsilon) \sum_{p \in P_1} \mathbf{d}(p, c_1^*) + 8\alpha(1 + \varepsilon) \sum_{p \in P_1} \mathbf{d}(p, c_1^*) + (1 + \varepsilon) \sum_{p \in P_2} \mathbf{d}(p, c_2^*) \\
&\leq (1 + 8\alpha)(1 + \varepsilon) \mathit{opt}_2(P)
\end{aligned}$$

Case 2: If there is no recursive call with $|P_2 \cap \bar{P}| \geq \alpha|\bar{P}|$ the pruning phase will be called recursively $\lceil \log n \rceil$ times until there is a single point $q \in \bar{P}$ which can be assigned to a cluster by itself with a cluster center $c_q \in C$ at a cost of $\mathbf{d}(q, c_q) \leq (1 + \varepsilon)\mathbf{d}(q, C)$. Then, $N = P \setminus \{q\}$ and the proof of the previous case also bounds the cost of assigning $P \cap N$ to \bar{c}_1 .

For $k > 2$, consider the algorithm as each center is added. Let $\bar{C}_i = \{\bar{c}_1, \dots, \bar{c}_i\}$ be the i medians already approximated, corresponding to supercluster P'_1 , with P'_2 consisting of clusters whose medians are yet to be found. Similar analysis as above shows that the cost of points incorrectly assigned to centers in \bar{C}_i in the pruning phases can be bounded by $8\alpha k \sum_{p \in P'_1} \mathbf{d}(p, \bar{C}_i)$. See the proof of Theorem 2.5 of [ABS10] for full details. \square

2.3.2 Running time of CLUSTER

We characterize the running time of CLUSTER in terms of \mathbf{d} , C and F , and certain *operations* involving them.

For any $\varepsilon' \geq 0$, a $(1 + \varepsilon')$ -approximate nearest neighbor or closest point in C for any $x \in X$ is a point $y \in C$ such that $\mathbf{d}(x, y) \leq (1 + \varepsilon')\mathbf{d}(x, C)$; note that $\varepsilon' = 0$ gives an exact nearest neighbor. Let $\mathcal{Q}(C)$ denote the maximum time required to compute a $(1 + \varepsilon')$ -approximate nearest neighbor in C for any $x \in X$ and $\varepsilon' \geq 0$; h depends on ε' , and the value of ε' itself depends on the context (and is usually $O(1)$ or $O(\varepsilon)$) but for simplicity we will ignore this dependence for now, and make it explicit later. Further, h is a non-increasing function of ε' . Let $\mathcal{D}(C)$ denote the maximum time required to compute $\mathbf{d}(x, y)$

for any $x \in X, y \in C$. Finally, let $w(m) = \max_{S \subseteq X, |S|=m} |F(S)|$, and let $f(m)$ be the maximum number of operations needed to compute $F(S)$ for any $S \subseteq X$ of size m , where computing d between points in X and C , and computing an approximate closest point in C to any point in X count as one operation each.

Remark. The previous conference version of this work [NT20] only considered $\varepsilon' = 0$; in this work we relax this restriction and show that computing an approximate nearest neighbor is sufficient, which allows us to get faster clustering algorithms in certain cases.

Lemma 5 Running time of CLUSTER. *Suppose (X, C, d) satisfies the strong or weak sampling properties (Definitions 1 and 2) for some $\varepsilon, \delta \in (0, 1)$. Given $P \subseteq X$ containing n points, algorithm CLUSTER runs in time*

$$n \cdot 2^{O(km_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot (w(m_{\delta,\varepsilon}) \cdot f(m_{\delta,\varepsilon}))^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C)).$$

Proof. The proof is similar to the running time analysis from [ABS10]. We will first count the number of operations required, where computing d between points in X and C , and computing an approximate closest point in C for any point in X both count as one operation each.

Let $T(n, k)$ denote the number of operations required by CLUSTER with n input points and k medians to be found. For $k = 0$, we clearly have $T(n, 0) = O(1)$. For $n \leq k$, we can put each input point in its own cluster, and return a $(1 + \varepsilon)$ -nearest neighbor in C for each input point as the cluster medians. Thus, $T(n, k) \leq O(n)$.

Let us consider the case $n > k \geq 1$. In the sampling phase, the number of candidate centers generated is $2^{O(m_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot w(m_{\delta,\varepsilon})$, each taking $f(m_{\delta,\varepsilon})$ operations. Each of the candidate centers is then tried recursively, each taking $T(n, k - 1)$ operations. The pruning phase takes $O(n)$ operations. After pruning, the algorithm is called once for the remaining point set, requiring $T(n/2, k)$ operations. We thus have

$$\begin{aligned} T(n, k) &\leq 2^{O(m_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot w(m_{\delta,\varepsilon}) \cdot (f(m_{\delta,\varepsilon}) + T(n, k - 1)) + T(n/2, k) + O(n) \\ &\leq 2^{O(m_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot w(m_{\delta,\varepsilon}) \cdot f(m_{\delta,\varepsilon}) \cdot T(n, k - 1) + T(n/2, k) + O(n). \end{aligned}$$

Solving the recurrence yields

$$T(n, k) = n \cdot 2^{O(km_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot (w(m_{\delta,\varepsilon}) \cdot f(m_{\delta,\varepsilon}))^{O(k)}.$$

Taking into account the time to compute \mathbf{d} and an approximate closest point in C , we get the total running time to be $n \cdot 2^{O(km_{\delta,\varepsilon} \log(\frac{1}{\alpha}m_{\delta,\varepsilon}))} \cdot (w(m_{\delta,\varepsilon}) \cdot f(m_{\delta,\varepsilon}))^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C))$. \square

By setting $\alpha = \frac{\varepsilon}{8k^2}$, the approximation factor in Lemma 4 becomes $(1 + 3\varepsilon)$. Moreover, the error probability can be made arbitrarily small by running the CLUSTER algorithm $2^{\Theta(k)}$ times and taking the minimum cost solution, without changing the asymptotic running time. We thus get the following. Note that $w(m_{\delta,\varepsilon})$ takes on values 1 and $t_{\delta,\varepsilon}$ for the strong and weak sampling properties respectively.

Theorem 6 CLUSTER with strong sampling. *Suppose (X, C, \mathbf{d}) satisfies the strong sampling property (Definition 1) for some $\varepsilon, \delta \in (0, 1)$. Further, suppose c_S can be computed in $\mathcal{A}(m_{\delta,\varepsilon})$ operations, where computing \mathbf{d} between points in X and C , and computing an approximate closest point in C to any point in X count as one operation each. Given $P \subseteq X$ having n points and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, a $(1 + 3\varepsilon)$ -approximate solution to the (k, C) -median problem for P can be computed in time*

$$n \cdot 2^{O(km_{\delta,\varepsilon} \log(\frac{k}{\varepsilon}m_{\delta,\varepsilon}))} \cdot \mathcal{A}(m_{\delta,\varepsilon})^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C)).$$

Theorem 7 CLUSTER with weak sampling. *Suppose (X, C, \mathbf{d}) satisfies the weak sampling property (Definition 2) for some $\varepsilon, \delta \in (0, 1)$. Further, suppose $\Gamma(S)$ can be computed in $\mathcal{G}(m_{\delta,\varepsilon})$ operations, where computing \mathbf{d} between points in X and C , and computing an approximate closest point in C to any point in X count as one operation each. Given $P \subseteq X$ having n points and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, a $(1 + 3\varepsilon)$ -approximate solution to the (k, C) -median problem for P can be computed in time*

$$n \cdot 2^{O(km_{\delta,\varepsilon} \log(\frac{k}{\varepsilon}m_{\delta,\varepsilon}))} \cdot (t_{\delta,\varepsilon} \cdot \mathcal{G}(m_{\delta,\varepsilon}))^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C)).$$

2.4 Covering metric spaces

We specify sufficient conditions on C for the strong and weak sampling properties to hold. These conditions characterize how well can certain subsets of C be *covered* using a small number of sets.

Let $\mathcal{X} = (X, \mathbf{d})$ be a metric space. Given $x \in X$, let $\mathcal{B}_{\mathbf{d}}(x, r) = \{x' \in X \mid \mathbf{d}(x, x') \leq r\}$ denote the ball of radius r (under \mathbf{d}) centered at x ; we will drop the subscript \mathbf{d} if it is clear from the context. An ***r-cover*** of a subset $X' \subseteq X$ for some $r > 0$ is a set $Y \subseteq X$ such that $X' \subseteq \bigcup_{y \in Y} \mathcal{B}(y, r)$. Note that the elements of Y need not be in X' . Also note that if Y is an r -cover for X' , it is also an r -cover for any subset of X' .

Suppose $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is a non-decreasing function that does not increase very rapidly. Formally, for any $\varepsilon, \delta \in (0, 1)$ there exists a sufficiently large m such that

$$g\left(\frac{30m}{\delta\varepsilon}\right) \exp\left(\frac{-\varepsilon^2 m}{144}\right) < \delta/2.$$

Then, a subset $Y \subseteq X$ is said to be ***g-coverable*** for such a function g iff for all $y \in Y$ and $r > r' > 0$, there exists an r' -cover of $\mathcal{B}(y, r) \cap Y$ of size at most $g(r/r')$. Note that if Y is g -coverable then any subset $Y' \subseteq Y$ is also g -coverable.

Intuitively, if C has a small cover, then for any metric ball in C there exists a small set of points (not necessarily from C), termed the cover, such that the distance from any point in the ball to a point in the cover is smaller than the radius of the ball.

2.4.1 Sufficient conditions for the strong sampling property

We give sufficient conditions for the strong sampling property to hold in terms of coverability of C . Note that the strong sampling property requires that the optimal 1-median can be computed efficiently, and usually does not hold for many common metric spaces such as the Euclidean metric (and by extension, the discrete Fréchet and Hausdorff metrics). The proof is similar to Lemma 3.4 of [ABS10] but has been adapted to our setting. It is worthwhile to note that the notion of coverability of C is more general than C having constant doubling dimension – in particular the cover of any subset of C need not come from C .

The following two useful lemmas hold for any metric space $\mathcal{X} = (X, \mathbf{d})$, and are restated for convenience.

Lemma 8 [ABS10], Lemma 3.2. *Let $c \in X$, $P \subseteq X$ of size n , and $\delta > 0$. A uniform sample multiset $S \subseteq P$ of size m satisfies*

$$\Pr \left[\exists q \in S \mid \mathbf{d}(q, c) \geq \frac{1}{\delta n} \sum_{p \in P} \mathbf{d}(p, c) \right] \leq m\delta.$$

Lemma 9 [ABS10], Lemma 3.3. *Let $\varepsilon \in (0, 1]$, $P \subseteq X$ of size n , and $b, c \in X$ be such that $\sum_{p \in P} \mathbf{d}(p, b) > (1 + \frac{4}{5}\varepsilon) \sum_{p \in P} \mathbf{d}(p, c)$. A uniform sample multiset $S \subseteq P$ of size m satisfies*

$$\Pr \left[\sum_{s \in S} \mathbf{d}(s, b) \leq \sum_{s \in S} \mathbf{d}(s, c) + \frac{\varepsilon m}{5n} \sum_{p \in P} \mathbf{d}(p, c) \right] < \exp \left(-\frac{\varepsilon^2 m}{144} \right).$$

Intuitively, Lemma 8 says that if we are sampling points uniformly, the probability of sampling a point far away from a fixed point is small and can be upper-bounded, but the bound increases with the size of the sample. Lemma 9 states that if the average distance of a point set to a point c is smaller than to a point b , then the average distance of a uniform sample of the point set to c relative to b is also small, with some probability that can be lower-bounded.

We now prove the main theorem.

Theorem 10 Sufficient conditions for strong sampling. *If C is g -coverable, and for any $P \subseteq X$, c_P can be computed in time depending only on $|P|$, then (X, C, \mathbf{d}) satisfies the strong sampling property (Definition 1) for any $\varepsilon, \delta \in (0, 1)$. Here, the constant $m_{\delta, \varepsilon} = m_{\delta, \varepsilon, g}$ also depends on g .*

Proof. Let $P, S, \varepsilon, \delta$ be as specified in Definition 1. Let $|P| = n$ and $|S| = m$; the value of m will be set later in the proof.

Let $r = \frac{6m}{\delta n} \sum_{p \in P} \mathbf{d}(p, c_P)$. Let $r' = \frac{r}{3}$, $U = \mathcal{B}(c_P, r)$, and $U' = \mathcal{B}(c_P, r')$. By Lemma 8 we have

$$\Pr \left[\exists s \in S \mid \mathbf{d}(c_P, s) > r' \right] = \Pr \left[\exists s \in S \mid \mathbf{d}(c_P, s) > \frac{2m}{\delta} \cdot \frac{1}{n} \sum_{p \in P} \mathbf{d}(c_P, p) \right] \leq \delta/2.$$

Thus, for all $s \in S$, $d(c_P, s) \leq r'$ with probability at least $1 - \delta/2$, and hence $S \subseteq U'$ with probability at least $1 - \delta/2$. Now if $c_S \notin U$, then $d(c_P, c_S) > r = 3r'$ and by triangle inequality

$$\sum_{s \in S} d(s, c_S) \geq \sum_{s \in S} (d(c_P, c_S) - d(c_P, s)) > \sum_{s \in S} 2r' = 2r'm.$$

However since $S \subseteq U'$, we have $\sum_{s \in S} d(s, c_P) \leq r'm < \sum_{s \in S} d(s, c_S)$, contradicting the claim that c_S is the optimal $(1, C)$ -median of S . Thus, with probability at least $1 - \delta/2$, $c_S \in U$.

Since C is g -coverable, there exists a $(\frac{\varepsilon}{5n} \sum_{p \in P} d(p, c_P))$ -cover of $U \cap C$ of size $g(\frac{30m}{\delta\varepsilon})$. Let C' be such a cover. Define

$$C'_{bad} = \{c \in C' \mid \sum_{p \in P} d(p, c) > (1 + \frac{4}{5}\varepsilon) \sum_{p \in P} d(p, c_P)\}.$$

Setting m to be the sufficiently large constant $m_{\delta, \varepsilon, g}$ and using the union bound and Lemma 9 we get

$$\begin{aligned} & \Pr \left[\exists c \in C'_{bad} \mid \sum_{s \in S} d(s, c) \leq \sum_{s \in S} d(s, c_P) + \frac{\varepsilon m}{5n} \sum_{p \in P} d(p, c_P) \right] \\ & < g \left(\frac{30m}{\delta\varepsilon} \right) \exp \left(\frac{-\varepsilon^2 m}{144} \right) \\ & < \delta/2. \end{aligned}$$

Thus, with probability at least $1 - \delta/2$, for all $c \in C'_{bad}$ we have

$$\sum_{s \in S} d(s, c) > \sum_{s \in S} d(s, c_P) + \frac{\varepsilon m}{5n} \sum_{p \in P} d(p, c_P).$$

Let c' be the closest point in C' to c_S . By definition of C' and the fact that $c_S \in U \cup C$, we have $d(c_S, c') \leq \frac{\varepsilon}{5n} \sum_{p \in P} d(p, c_P)$. We then have

$$\begin{aligned} \sum_{s \in S} d(s, c') & \leq \sum_{s \in S} (d(s, c_S) + d(c_S, c')) \leq \sum_{s \in S} d(s, c_S) + \frac{\varepsilon m}{5n} \sum_{p \in P} d(p, c_P) \\ & \leq \sum_{s \in S} d(s, c_P) + \frac{\varepsilon m}{5n} \sum_{p \in P} d(p, c_P) < \sum_{s \in S} d(s, c) \end{aligned}$$

for all $c \in C'_{bad}$ from previous inequality. Thus $c' \notin C'_{bad}$ and hence

$$\sum_{p \in P} \mathbf{d}(p, c') \leq (1 + \frac{4}{5}\varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P).$$

We then conclude

$$\sum_{p \in P} \mathbf{d}(p, c_S) \leq \sum_{p \in P} \mathbf{d}(p, c') + n\mathbf{d}(c', c_S) \leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P).$$

This event holds with probability at least $(1 - \delta/2)^2 > 1 - \delta$. \square

From Theorems 6 and 10, we get the following.

Corollary 11 Clustering with coverability and strong sampling. *Suppose C is g -coverable and the optimal $(1, C)$ -median of any subset of X can be computed in time depending on the size of the subset. Let $\varepsilon, \delta \in (0, 1)$. Given $P \subseteq X$ having n points and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, a $(1 + 3\varepsilon)$ -approximate solution to the (k, C) -median problem for P can be computed in time*

$$n \cdot 2^{O(km_{\delta, \varepsilon, g} \log(\frac{k}{\varepsilon} m_{\delta, \varepsilon, g}))} \cdot \mathcal{A}(m_{\delta, \varepsilon, g})^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C)),$$

where $m_{\delta, \varepsilon, g}$ is a constant depending only on ε, δ, g , and $\mathcal{A}(m)$ is the number of operations needed to compute the optimal $(1, C)$ -median of m points in X , where computing \mathbf{d} between points in X and C , and computing an approximate closest point in C to any point in X count as one operation each.

2.4.2 Sufficient conditions for the weak sampling property

We give sufficient conditions for the weak sampling property to hold in terms of coverability of C .

For $Y \subseteq X$, let $\mathcal{R}_Y(\frac{r}{r'})$ be the number of operations required to compute an r' -cover of $\mathcal{B}(y, r) \cap Y$ for any $y \in Y$ (if such a cover exists) where computing \mathbf{d} between points in X and C , and computing an approximate closest point in C to any point in X count as one operation each (we assume that the number of operations can be expressed in terms of $\frac{r}{r'}$).

COMPUTE-Q($P, C, \varepsilon, \delta, a, b$):

input: Point sets $P, C \subseteq X$; $\varepsilon, \delta \in (0, 1)$; and $a, b > 0$

$\varepsilon_1 \leftarrow 1, \varepsilon_2 \leftarrow \frac{\varepsilon}{3}, \varepsilon_3 \leftarrow 1$

$q \leftarrow$ chosen uniformly at random from P

$C' \leftarrow (\varepsilon_2 a)$ -cover of $\mathcal{B}\left(q'_{\varepsilon_1}, \frac{(2+\varepsilon_1)b}{\delta}\right) \cap C$, where q'_{ε_1} is a $(1 + \varepsilon_1)$ -nearest neighbor of q in C

$Q \leftarrow \{c'_{\varepsilon_3} \mid c \in C'\}$ where c'_{ε_3} is a $(1 + \varepsilon_3)$ -nearest neighbor of c in C

return Q

Figure 2.3: Algorithm COMPUTE-Q.

The following lemma will be helpful, and states that if C has a small cover and if we have a good estimate of the *cost* of the optimal $(1, C)$ -median, then the algorithm COMPUTE-Q (Fig. 2.3) computes a small set of points in $Q \subseteq C$ such that at least one of them is a good approximation to the optimal $(1, C)$ -median. Further, this can be done in time independent of $|P|$. Both of these properties are necessary for the weak sampling property (Definition 2).

Lemma 12. *Let $P \subseteq X$ and $\varepsilon, \delta \in (0, 1)$. Suppose C is g -coverable. Then given a, b such that $a \leq \frac{1}{|P|} \sum_{p \in P} \mathbf{d}(c_P, p) \leq b$, algorithm COMPUTE-Q computes a set $Q \subseteq C$ of size $O(g(\frac{9b}{\varepsilon\delta a}))$ such that*

$$\Pr \left[\exists q \in Q \mid \sum_{p \in P} \mathbf{d}(p, q) \leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P) \right] \geq 1 - \delta.$$

Further, COMPUTE-Q takes $O(\mathcal{R}_C(\frac{9b}{\varepsilon\delta a}) + g(\frac{9b}{\varepsilon\delta a}))$ operations, where computing \mathbf{d} between points in X and C , and computing a $(1 + \frac{\varepsilon}{3})$ -approximate closest point in C to any point in X count as one operation each.

Proof. For any $x \in X$, we define $x' = \operatorname{argmin}_{y \in C} \mathbf{d}(x, y)$. Further, for any $\varepsilon' > 0$, define $x'_{\varepsilon'}$ to be a $(1 + \varepsilon')$ -nearest neighbor of $x \in X$ in C .

Let $\varepsilon_1 = 1, \varepsilon_2 = \frac{\varepsilon}{3}$, and $\varepsilon_3 = 1$, as used in COMPUTE-Q.

Consider $q \in P$ chosen uniformly at random. By Markov's inequality, $\mathbf{d}(q, c_P) \leq$

$\frac{1}{\delta|P|} \sum_{p \in P} \mathbf{d}(p, c_P)$ with probability $\geq 1 - \delta$. In such a case,

$$\begin{aligned} \mathbf{d}(q'_{\varepsilon_1}, c_P) &\leq \mathbf{d}(q'_{\varepsilon_1}, q) + \mathbf{d}(q, c_P) \leq (1 + \varepsilon_1)\mathbf{d}(q', q) + \mathbf{d}(q, c_P) \\ &\leq (2 + \varepsilon_1)\mathbf{d}(q, c_P) \\ &\leq \frac{2 + \varepsilon_1}{\delta|P|} \sum_{p \in P} \mathbf{d}(p, c_P). \end{aligned}$$

Thus, $c_P \in \mathcal{B}\left(q'_{\varepsilon_1}, \frac{(2+\varepsilon_1)b}{\delta}\right)$ with probability at least $1 - \delta$.

Let C' be an $(\varepsilon_2 a)$ -cover of $\mathcal{B}\left(q'_{\varepsilon_1}, \frac{(2+\varepsilon_1)b}{\delta}\right) \cap C$. Since C is g -coverable, $|C'| = g\left(\frac{(2+\varepsilon_1)b}{\varepsilon_2 \delta a}\right) = g\left(\frac{9b}{\varepsilon \delta a}\right)$. We will argue that $Q = \{c'_{\varepsilon_3} \mid c \in C'\}$ returned by COMPUTE-Q satisfies the inequality stated in the lemma. Let $x = \operatorname{argmin}_{y \in C'} \mathbf{d}(y, c_P)$. Since C' is an $(\varepsilon_2 a)$ -cover, $\mathbf{d}(x, c_P) \leq \varepsilon_2 a$. Also,

$$\begin{aligned} \mathbf{d}(x'_{\varepsilon_3}, c_P) &\leq \mathbf{d}(x'_{\varepsilon_3}, x) + \mathbf{d}(x, c_P) \leq (1 + \varepsilon_3)\mathbf{d}(x, x') + \varepsilon_2 a \\ &\leq (1 + \varepsilon_3)\mathbf{d}(x, c_P) + \varepsilon_2 a \\ &\leq (2 + \varepsilon_3)\varepsilon_2 a. \end{aligned}$$

We then have

$$\begin{aligned} \sum_{p \in P} \mathbf{d}(p, x'_{\varepsilon_3}) &\leq \sum_{p \in P} (\mathbf{d}(p, c_P) + \mathbf{d}(x'_{\varepsilon_3}, c_P)) \leq \left(\sum_{p \in P} \mathbf{d}(p, c_P) \right) + (2 + \varepsilon_3)\varepsilon_2 a|P| \\ &\leq (1 + (2 + \varepsilon_3)\varepsilon_2) \sum_{p \in P} \mathbf{d}(p, c_P) \\ &\leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P), \end{aligned}$$

by our choice of ε_2 and ε_3 .

Computing C' takes $\mathcal{R}_C\left(\frac{9b}{\varepsilon \delta a}\right)$ operations. Computing the output set takes $|C'| = O\left(g\left(\frac{9b}{\varepsilon \delta a}\right)\right)$ operations. \square

The next theorem shows that with a small random sample of P , one of two things can happen. Either one of the samples is close to an approximate $(1, C)$ -median, or we can approximate the cost of the optimal $(1, C)$ -median in time independent of $|P|$. This along with Lemma 12 shows that the weak sampling property holds if C is g -coverable. The

COMPUTE- $\Gamma(P, C, \varepsilon, \delta)$:
input: Point sets $P, C \subseteq X$; $1 < \varepsilon < \frac{4}{9}$ and $(1 - \frac{5}{18}\varepsilon) < \delta < 1$
 $\varepsilon_1 \leftarrow \frac{\varepsilon}{4}, \varepsilon_2 \leftarrow 1$
 $Q \leftarrow$ uniform random multiset of P of size $\frac{1}{\varepsilon_1}$
 $q \leftarrow$ chosen uniformly at random from P
 $q'_{\varepsilon_2} \leftarrow (1 + \varepsilon_2)$ -nearest neighbor of q in C
 $a \leftarrow \left(\frac{\varepsilon_1^3}{2 + \varepsilon_2}\right) \sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2})$
 $b \leftarrow \frac{1}{\varepsilon_1} \sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2})$
 $Q_1 \leftarrow$ COMPUTE-Q($P, C, \varepsilon, \delta_1, a, b$), where $\delta_1 \leftarrow 2\varepsilon_1 - \frac{9}{5}(1 - \delta)$
return $Q_1 \cup \{q'_{\varepsilon_2}\}$

Figure 2.4: Algorithm COMPUTE- Γ .

proof is inspired by the proof of Theorem 1 in [KSS05], and it also shows that the algorithm COMPUTE- Γ (Fig. 2.4) computes Γ for the weak sampling property (Definition 2).

Theorem 13 Sufficient conditions for weak sampling. *If C is g -coverable, then (X, C, \mathbf{d}) satisfies the weak sampling property (Definition 2) for $0 < \varepsilon < \frac{4}{9}$ and $(1 - \frac{5}{18}\varepsilon) < \delta < 1$. Further, the constants $m_{\delta, \varepsilon} = 1 + \frac{4}{\varepsilon}$ and $t_{\delta, \varepsilon} = O\left(g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$; and algorithm COMPUTE- Γ computes $\Gamma(S)$ in $O\left(\frac{1}{\varepsilon} + \mathcal{R}_C\left(\frac{6912}{\delta_1 \varepsilon^5}\right) + g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$ number of operations, where $\delta_1 = \frac{\varepsilon}{2} - \frac{9}{5}(1 - \delta)$, and computing \mathbf{d} between points in X and C , and computing a $(1 + \frac{\varepsilon}{3})$ -approximate closest point in C to any point in X count as one operation each.*

Proof. Let $\varepsilon_1 = \frac{\varepsilon}{4}$, $\varepsilon_2 = 1$ and $\bar{r} = \frac{1}{|P|} \sum_{p \in P} \mathbf{d}(p, c_P)$. Also, let $x' = \operatorname{argmin}_{y \in C} \mathbf{d}(x, y)$ for any $x \in X$. Further, for any $\varepsilon' > 0$, define $x'_{\varepsilon'}$ to be a $(1 + \varepsilon')$ -nearest neighbor of $x \in X$ in C .

Let $Q \subseteq P$ be a uniform random multiset of size $\frac{1}{\varepsilon_1}$, and $q \in P$ be another point chosen uniformly at random. We will show that $Q \cup \{q\}$ plays the role of S in Definition 2.

Using Markov's inequality and union bound, we have

$$\Pr\left[\mathbf{d}(q, c_P) > \frac{\bar{r}}{2\varepsilon_1^2}\right] < 2\varepsilon_1^2 \text{ and } \Pr\left[\exists p \in Q \mid \mathbf{d}(p, c_P) > \frac{\bar{r}}{2\varepsilon_1^2}\right] < \left(\frac{1}{\varepsilon_1}\right) 2\varepsilon_1^2 = 2\varepsilon_1.$$

Thus with probability $\geq 1 - 2\varepsilon_1 - 2\varepsilon_1^2$, q and Q are in $\mathcal{B}\left(c_P, \frac{\bar{r}}{2\varepsilon_1^2}\right)$. We assume that this

event happens. Now, by definition of q' and q'_{ε_2} , we have $\mathbf{d}(q, q'_{\varepsilon_2}) \leq (1 + \varepsilon_2)\mathbf{d}(q, c_P)$. Hence,

$$\mathbf{d}(q'_{\varepsilon_2}, c_P) \leq \mathbf{d}(q, q'_{\varepsilon_2}) + \mathbf{d}(q, c_P) \leq (2 + \varepsilon_2)\mathbf{d}(q, c_P) \leq \frac{(2 + \varepsilon_2)\bar{r}}{2\varepsilon_1^2}.$$

Let $\mathcal{B}_1 = \mathcal{B}\left(c_P, \frac{(2 + \varepsilon_2)\bar{r}}{2\varepsilon_1^2}\right)$, $\mathcal{B}_2 = \mathcal{B}(q'_{\varepsilon_2}, \varepsilon_1\bar{r})$ and $P' = P \cap \mathcal{B}_1$. Then, $q'_{\varepsilon_2} \in \mathcal{B}_1$ and $Q \subseteq P'$. We consider two cases now.

Case 1: P' has at least $2\varepsilon_1|P'|$ points outside \mathcal{B}_2 . For any $p \in Q$, the probability p is outside \mathcal{B}_2 is $\geq 2\varepsilon_1$. Thus, with probability at least $2\varepsilon_1$, there exists $p \in Q$ such that $\mathbf{d}(p, q'_{\varepsilon_2}) \geq \varepsilon_1\bar{r}$ and hence $\sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2}) \geq \varepsilon_1\bar{r}$. Also, $\mathbf{d}(p, q'_{\varepsilon_2}) \leq \mathbf{d}(p, c_P) + \mathbf{d}(q'_{\varepsilon_2}, c_P) \leq \left(\frac{2 + \varepsilon_2}{\varepsilon_1^2}\right)\bar{r}$, for any $p \in Q$. Hence, $\sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2}) \leq \left(\frac{2 + \varepsilon_2}{\varepsilon_1^2}\right)\bar{r}$.

Let $\delta_1 = 2\varepsilon_1 - \frac{9}{5}(1 - \delta)$. We can now use Lemma 12 with $a = \left(\frac{\varepsilon_1^3}{2 + \varepsilon_2}\right)\sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2})$ and $b = \frac{1}{\varepsilon_1}\sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2})$ to compute a set $Q_1 \subseteq C$, using $Q_1 = \text{COMPUTE-Q}(P, C, \varepsilon, \delta_1, a, b)$ with $|Q_1| = O\left(g\left(\frac{9b}{\varepsilon\delta_1 a}\right)\right) = O\left(g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$ candidate centers, one of which is a $(1 + \varepsilon)$ -approximate center with probability at least $1 - \delta_1$. The total probability of getting a good set of candidate centers is $(2\varepsilon_1 - \delta_1)(1 - 2\varepsilon_1 - 2\varepsilon_1^2) > 1 - \delta$.

Case 2: P' has at most $2\varepsilon_1|P'|$ points outside \mathcal{B}_2 . We further consider two cases.

Case 2(a): $\mathbf{d}(q'_{\varepsilon_2}, c_P) \leq 4\varepsilon_1\bar{r}$. Then

$$\sum_{p \in P} \mathbf{d}(p, q'_{\varepsilon_2}) \leq \sum_{p \in P} (\mathbf{d}(p, c_P) + \mathbf{d}(q'_{\varepsilon_2}, c_P)) \leq (1 + 4\varepsilon_1) \sum_{p \in P} \mathbf{d}(p, c_P) \leq (1 + \varepsilon) \sum_{p \in P} \mathbf{d}(p, c_P).$$

Thus, $\{q'_{\varepsilon_2}\}$ is the set of candidates.

Case 2(b): $\mathbf{d}(q'_{\varepsilon_2}, c_P) > 4\varepsilon_1\bar{r}$. Let $\alpha = \frac{2\varepsilon_1^2}{2 + \varepsilon_2}$. By our choice of ε_1 and ε_2 , $\alpha \in (0, 1)$. Also, by definition of P' , $\mathbf{d}(p, c_P) \geq \frac{\bar{r}}{\alpha}$ for all $p \in P \setminus P'$. We then have

$$\begin{aligned} \bar{r} \cdot |P| &\geq \sum_{p \in P \setminus P'} \mathbf{d}(p, c_P) \geq \frac{\bar{r}}{\alpha} \cdot (|P| - |P'|) \\ \Rightarrow |P'| &\geq (1 - \alpha)|P|. \end{aligned}$$

Hence, we have $|P'| \geq \left(1 - \frac{2\varepsilon_1^2}{2 + \varepsilon_2}\right)|P|$. Then, the number of points of P that are outside

\mathcal{B}_2 is at most

$$\begin{aligned} |P \setminus P'| + 2\varepsilon_1|P'| &= |P| - (1 - 2\varepsilon_1)|P'| \\ &\leq |P| - (1 - 2\varepsilon_1) \left(1 - \frac{2\varepsilon_1^2}{2 + \varepsilon_2}\right) |P| \\ &= \left(1 - (1 - 2\varepsilon_1) \left(1 - \frac{2\varepsilon_1^2}{2 + \varepsilon_2}\right)\right) |P|. \end{aligned}$$

Thus, $|P \cap \mathcal{B}_2| \geq (1 - 2\varepsilon_1) \left(1 - \frac{2\varepsilon_1^2}{2 + \varepsilon_2}\right) |P|$. Now, suppose we assign all points from c_P to q'_{ε_2} . For $p \in P \cap \mathcal{B}_2$, the decrease in cost on switching from c_P to q'_{ε_2} is at least $\mathbf{d}(p, c_P) - \mathbf{d}(p, q'_{\varepsilon_2})$. By definition of \mathcal{B}_2 , we have $\mathbf{d}(p, q'_{\varepsilon_2}) \leq \varepsilon_1 \bar{r}$. Also by triangle inequality, $\mathbf{d}(p, c_P) \geq \mathbf{d}(q'_{\varepsilon_2}, c_P) - \mathbf{d}(p, q'_{\varepsilon_2})$. We can thus lower bound the decrease in cost for $p \in P \cap \mathcal{B}_2$ by

$$\mathbf{d}(p, c_P) - \mathbf{d}(p, q'_{\varepsilon_2}) \geq \mathbf{d}(q'_{\varepsilon_2}, c_P) - 2\mathbf{d}(p, q'_{\varepsilon_2}) \geq \mathbf{d}(q'_{\varepsilon_2}, c_P) - 2\varepsilon_1 \bar{r}.$$

For $p \in P \setminus \mathcal{B}_2$, the increase in cost on switching from c_P to q'_{ε_2} is at most

$$\mathbf{d}(p, q'_{\varepsilon_2}) - \mathbf{d}(p, c_P) \leq \mathbf{d}(q'_{\varepsilon_2}, c_P).$$

The overall decrease in cost is

$$\begin{aligned} &|P \cap \mathcal{B}_2|(\mathbf{d}(q'_{\varepsilon_2}, c_P) - 2\varepsilon_1 \bar{r}) - |P \setminus \mathcal{B}_2|\mathbf{d}(q'_{\varepsilon_2}, c_P) \\ &> |P \cap \mathcal{B}_2| \frac{\mathbf{d}(q'_{\varepsilon_2}, c_P)}{2} - |P \setminus \mathcal{B}_2|\mathbf{d}(q'_{\varepsilon_2}, c_P). \end{aligned}$$

by our assumption that $\mathbf{d}(q'_{\varepsilon_2}, c_P) > 4\varepsilon_1 \bar{r}$. Hence, the decrease in cost will be positive if

$$\frac{|P \cap \mathcal{B}_2|}{2} - |P \setminus \mathcal{B}_2| = \frac{|P \cap \mathcal{B}_2|}{2} - (|P| - |P \cap \mathcal{B}_2|) = \frac{3}{2}|P \cap \mathcal{B}_2| - |P| \geq 0,$$

which can be shown from our earlier bound $|P \cap \mathcal{B}_2| \geq (1 - 2\varepsilon_1) \left(1 - \frac{2\varepsilon_1^2}{2 + \varepsilon_2}\right) |P|$ and our choice of $\varepsilon_1, \varepsilon_2$. But c_P is the optimal $(1, C)$ -median of P , a contradiction. Hence case 2(b) cannot occur.

From the two cases above, we have shown that $Q_1 \cup \{q'_{\varepsilon_2}\}$ plays the role of $\Gamma(S)$ in Definition 2, and this is the exact set returned by COMPUTE- Γ .

Sampling Q, q takes time $O(\frac{1}{\varepsilon})$. Computing $\sum_{p \in Q} \mathbf{d}(p, q'_{\varepsilon_2})$ involves $|Q| = O(\frac{1}{\varepsilon})$ computations of \mathbf{d} between a pair of points from X , at least one of which comes from C . Computing

the set of candidates takes $O\left(\mathcal{R}_C\left(\frac{6912}{\delta_1 \varepsilon^5}\right) + g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$ operations. Thus, total number of operations needed is $O\left(\frac{1}{\varepsilon} + \mathcal{R}_C\left(\frac{6912}{\delta_1 \varepsilon^5}\right) + g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$. Moreover, $m_{\delta,\varepsilon} = |Q \cup \{q\}| = 1 + \frac{1}{\varepsilon} = 1 + \frac{4}{\varepsilon}$, and $t_{\delta,\varepsilon} = |Q_1 \cup \{q'_{\varepsilon_2}\}| = O\left(g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$. \square

From Theorems 7 and 13, we get the following.

Corollary 14 Clustering with coverability and weak sampling. *Suppose C is g -coverable. Let $\varepsilon \in (0, \frac{4}{9})$, $\delta \in (1 - \frac{5}{18}\varepsilon, 1)$. Given $P \subseteq X$ of size n and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, a $(1 + 3\varepsilon)$ -approximate solution to the (k, C) -median problem for P can be computed in time*

$$n \cdot 2^{O\left(\frac{k}{\varepsilon} \log\left(\frac{k}{\varepsilon}\right)\right)} \cdot (\mathcal{G}(m_{\delta,\varepsilon}) \cdot t_{\delta,\varepsilon})^{O(k)} \cdot (\mathcal{Q}(C) + \mathcal{D}(C)),$$

where $\mathcal{G}(m_{\delta,\varepsilon}) = O\left(\frac{1}{\varepsilon} + \mathcal{R}_C\left(\frac{6912}{\delta_1 \varepsilon^5}\right) + g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$, $t_{\delta,\varepsilon} = O\left(g\left(\frac{6912}{\delta_1 \varepsilon^5}\right)\right)$, and $\delta_1 = \frac{\varepsilon}{2} - \frac{9}{5}(1 - \delta)$. Here, $h(C)$ denotes the time required to compute a $(1 + \frac{\varepsilon}{3})$ -approximate nearest neighbor in C for any point in X , and $t(C)$ denotes the time required to compute $\mathfrak{d}(x, y)$ for any $x \in X, y \in C$.

2.5 Clustering discrete Fréchet and Hausdorff distances

In this section, we show how the results from the previous section can be used to cluster trajectories and point sets under the discrete Fréchet and Hausdorff distances respectively.

Clustering under discrete Fréchet distance. Recall that T^l is the set of all trajectories in \mathbb{R}^d having at most l points each; thus $T = \bigcup_{l>0} T^l$ is the set of all trajectories in \mathbb{R}^d . Given $\mathcal{T} = (T, \mathfrak{d}_F)$ and trajectories $P \subseteq T$, the (k, l) -median problem [DKS16, BDS19] is equivalent to the (k, T^l) -median problem in our setting, i.e., the center trajectories contain at most l points. We show that T^l is g -coverable for some g that depends on l .

Lemma 15 Coverability under discrete Fréchet distance. *T^l is g -coverable under \mathfrak{d}_F for $g(x) = l^{2l} \cdot x^{O(dl)}$. Further, an r' -cover of $\mathcal{B}_{\mathfrak{d}_F}(\gamma, r) \cap T^l$ for $r > r' > 0$ and $\gamma \in T^l$ can be computed in $l^{2l} \cdot \left(\frac{r}{r'}\right)^{O(dl)}$ time.*

Proof. Let $r > r' > 0$ be arbitrary. Let $\gamma = \langle p_1, \dots, p_{l'} \rangle \in T^l$ for some $l' \leq l$. Since the Euclidean metric in \mathbb{R}^d has doubling dimension $O(d)$, for any $p \in \mathbb{R}^d$ there exist $\left(\frac{r}{r'}\right)^{O(d)}$ points in the Euclidean ball $\mathcal{B}_E(p, r)$ centered at p such that any point in $\mathcal{B}_E(p, r)$ is at most r' distance away from one of these points; denote these points by $B_p(r, r')$. Consider the set of points $\bigcup_{i=1}^{l'} B_{p_i}(r, r')$; this set has cardinality $l' \cdot \left(\frac{r}{r'}\right)^{O(d)}$.

Next, consider the set of all trajectories T' defined by at most $2l$ points from $\bigcup_{i=1}^{l'} B_{p_i}(r, r')$ and containing at least one point from $B_{p_i}(r, r')$ for every i ; further these points respect the ordering of the sets that they belong to, i.e., if $p \in B_{p_i}(r, r')$, $q \in B_{p_j}(r, r')$, and $i < j$, then p appears before q in the trajectory (for points coming from the same set $B_{p_i}(r, r')$ all possible orderings are considered). Note that $|T'| \leq \left(l' \cdot \left(\frac{r}{r'}\right)^{O(d)}\right)^{2l}$. Further, $\mathbf{d}_F(\gamma, \gamma') \leq r$ for all $\gamma' \in T'$.

We will show that for any $\gamma'' \in \mathcal{B}_{\mathbf{d}_F}(\gamma, r) \cap T^l$, there exists $\gamma' \in T'$ such that $\mathbf{d}_F(\gamma', \gamma'') \leq r'$. Thus T' is the desired cover, completing the first part of our proof. Let $\gamma'' = \langle q_1, \dots, q_{l''} \rangle$ for some $l'' \leq l$. By definition of \mathbf{d}_F and the fact that $\mathbf{d}_F(\gamma, \gamma'') \leq r$, each q_i has a corresponding sequence of points $\langle p_{j_i}, p_{j_i+1}, \dots, p_{j'_i} \rangle$ each of which is at most r distance away from q_i . Moreover, for all $1 \leq i < l''$ we have $j'_i \leq j_{i+1} \leq j'_i + 1$, and $j_1 = 1, j'_{l''} = l'$.

For each q_i and $j \in \{j_i, j_i + 1, \dots, j'_i\}$, let u_j denote the point in $B_{p_j}(r, r')$ that is closest to q_i . Note that $q_i \in \mathcal{B}_E(p_j, r)$ and $\|q_i - u_j\| \leq r'$. Consider the sequence of points $\gamma(q_i) = \langle u_{j_i}, u_{j_i+1}, \dots, u_{j'_i} \rangle$. Then, $\mathbf{d}_F(\langle q_i \rangle, \gamma(q_i)) \leq r'$. Let γ' be the trajectory obtained by concatenating $\gamma(q_1), \dots, \gamma(q_{l''})$ in order. Then we get $\mathbf{d}_F(\gamma', \gamma'') \leq r'$. Further, by construction $\gamma' \in T'$.

As far as running time is concerned, computing the set $\bigcup_{i=1}^{l'} B_{p_i}(r, r')$ takes time $l' \cdot \left(\frac{r}{r'}\right)^{O(d)}$. From this set, computing T' takes time $l^{2l} \cdot \left(\frac{r}{r'}\right)^{O(dl)}$. \square

For a trajectory having m points, computing \mathbf{d}_F to any trajectory in T^l takes time $O(ml)$ using the standard dynamic programming algorithm, whereas computing the closest trajectory in T^l under \mathbf{d}_F takes $O\left(lm \log m \log\left(\frac{m}{l}\right)\right)$ time (see [BJW⁺08], Theorem 3). This, along with Corollary 14 and Lemma 15 give the following.

Theorem 16 Clustering under discrete Fréchet distance. *Let $\varepsilon \in (0, \frac{4}{9}), \delta \in (1 - \frac{5}{18}\varepsilon, 1)$. Given a set of n trajectories $P \subseteq T$ each having at most m points and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, the algorithm CLUSTER (Fig. 2.2) computes a $(1 + 3\varepsilon)$ -approximate solution to the (k, T^l) -median problem for P under the discrete Fréchet distance in time*

$$nm \log m \log \left(\frac{m}{l} \right) \cdot 2^{O(\frac{k}{\varepsilon} \log(\frac{k}{\varepsilon}))} \cdot \left(\frac{l}{\delta_1 \varepsilon} \right)^{O(kdl)},$$

where $\delta_1 = \frac{\varepsilon}{2} - \frac{9}{5}(1 - \delta)$.

Clustering under Hausdorff distance. Recall that U^l is the set of all point sets in \mathbb{R}^d containing at most l points each. Thus, $U = \bigcup_{l>0} U^l$ is the set of all finite point sets of \mathbb{R}^d . Given $\mathcal{U} = (U, \mathbf{d}_H)$ and subsets $P \subseteq U$, we show how to approximately solve the (k, U^l) -clustering problem for P and $k, l > 0$.

Lemma 17 Coverability under Hausdorff distance. *U^l is g -coverable under \mathbf{d}_H for $g(x) = l^l \cdot x^{O(dl)}$. Further, an r' -cover of $\mathcal{B}_{\mathbf{d}_H}(\zeta, r) \cap U^l$ for $r > r' > 0$ and $\zeta \in U^l$ can be computed in $l^l \cdot (\frac{r}{r'})^{O(dl)}$ time.*

Proof. Let $r > r' > 0$ be arbitrary, and let $\zeta = \{p_1, p_2, \dots, p_{l'}\}$ for some $l' \leq l$. Since the Euclidean metric has doubling dimension $O(d)$, there exist $(\frac{r}{r'})^{O(d)}$ points in the Euclidean ball $\mathcal{B}_E(p, r)$ such that any point in $\mathcal{B}_E(p, r)$ is at most r' distance away from one of these points; denote these points by $B_p(r, r')$.

Consider all subsets of size l of the set $\bigcup_{i=1}^{l'} B_{p_i}(r, r')$, denote this set by U' . Note that $|U'| = \left(l' \cdot \left(\frac{r}{r'} \right)^{O(d)} \right)^l$.

Next, consider any point set $\zeta' \in \mathcal{B}_{\mathbf{d}_H}(\zeta, r) \cap U^l$. By definition of Hausdorff distance, the points of ζ' (there are at most l of them) must lie in $\bigcup_{i=1}^{l'} \mathcal{B}_E(p_i, r)$. Thus, for each $p \in \zeta'$, there is some $i \in \{1, \dots, l'\}$ such that $\|p - q\| \leq r'$ for some $q \in B_{p_i}(r, r')$. Thus, there exists $\zeta'' \in U'$ such that $\mathbf{d}_H(\zeta', \zeta'') \leq r'$. Then, U' is the desired cover, completing the first part of our proof.

Computing $\bigcup_{i=1}^{l'} B_{p_i}(r, r')$ takes time $l' \cdot \left(\frac{r}{r'} \right)^{O(d)}$. Computing U' from it takes time $\left(l' \cdot \left(\frac{r}{r'} \right)^{O(d)} \right)^l$. □

Computing \mathbf{d}_H between two point sets of size m_1 and m_2 in \mathbb{R}^d takes time $O(m_1 m_2)$. Given $\zeta \in U$ of size m , computing the closest point to it in U^l (under \mathbf{d}_H) boils down to finding l disks in \mathbb{R}^d of minimum radius such that all the points of ζ lie inside the union of these disks; the centers of these disks give the desired set in U^l . This is the l -center problem in \mathbb{R}^d for the Euclidean metric. Solving this problem is NP-HARD when l is part of the input, and the brute force exact algorithm proceeds as follows. Note that the total number of subsets of ζ induced by disks in \mathbb{R}^d is $m^{O(d)}$. By looking at l such subsets at a time, we can pick the one that covers ζ and minimizes the radius of the largest disk; this takes total time $m^{O(dl)}$, and results in a factor of $nm^{O(dl)}$ in the final running time of the clustering algorithm [NT20].

Fortunately by Corollary 14, a $(1 + \frac{\varepsilon}{3})$ -approximate nearest neighbor in U^l for $\zeta \in U$ suffices; this means that the Euclidean l -center problem in the preceding paragraph has to be solved only approximately to within a factor of $(1 + \frac{\varepsilon}{3})$ of the optimal, and can be done in time $O(m \log l) + (\frac{l}{\varepsilon})^{O(l^{1-1/d})}$ [AP02]. This along with Corollary 14 and Lemma 17 give the following much faster running time.

Theorem 18 Clustering under Hausdorff distance. *Let $\varepsilon \in (0, \frac{4}{9})$, $\delta \in (1 - \frac{5}{18}\varepsilon, 1)$. Given a set of n point sets $P \subseteq U$ each having at most m points and $k \in \mathbb{N}$, with probability $\geq 1 - \delta$, the algorithm CLUSTER (Fig. 2.2) computes a $(1 + 3\varepsilon)$ -approximate solution to the (k, U^l) -median problem for P under the Hausdorff distance can be computed in time*

$$nm \cdot 2^{O(\frac{k}{\varepsilon} \log(\frac{k}{\varepsilon}))} \cdot \left(\frac{l}{\delta_1 \varepsilon}\right)^{O(kdl)},$$

where $\delta_1 = \frac{\varepsilon}{2} - \frac{9}{5}(1 - \delta)$.

2.6 Hardness and unbounded doubling dimension

In this section, we show a few negative results.

2.6.1 Hardness of clustering under the Hausdorff distance

We prove the following hardness result for k -median clustering under the Hausdorff distance.

Theorem 19. *The k -median clustering problem for finite point sets under the Hausdorff distance is NP-HARD.*

Proof. We reduce the Euclidean k -median problem, which is known to be NP-HARD [MS84]. The reduction is fairly straightforward – for each input point p of an instance of the Euclidean k -median problem, we have a singleton set $\{p\}$ as input to the Hausdorff k -median problem. Any solution to the Euclidean k -median problem is also a solution to the Hausdorff k -median problem of the same cost – we replace cluster center c in the Euclidean version by the cluster center $\{c\}$ for the Hausdorff version, and for each p assigned to c , we assign $\{p\}$ to $\{c\}$.

On the other hand, consider a solution to the instance of Hausdorff k -median problem. In particular, let S be a cluster center that is assigned the sets $\{\{p_1\}, \dots, \{p_n\}\}$. The cost of this single cluster is

$$\sum_{i=1}^n \mathbf{d}_H(\{p_i\}, S) = \sum_{i=1}^n \max_{s \in S} \|s - p_i\|,$$

by the definition of \mathbf{d}_H . Thus, replacing S by a singleton set $\{s\}$ for any $s \in S$ does not increase the cost of clustering. Hence we can assume that all cluster centers are singleton sets. We can then construct a solution for the Euclidean k -median problem by assigning p to s , where $\{s\}$ is the cluster center that $\{p\}$ was assigned to in the Hausdorff clustering solution. This does not increase the cost of the clustering as well. \square

2.6.2 Doubling dimension of discrete Fréchet and Hausdorff distances

We show that the discrete Fréchet distance does not have a doubling dimension bounded by a constant. The Hausdorff case can be shown similarly.

Suppose $\mathcal{T} = (T, \mathbf{d}_F)$ has a constant doubling dimension D . Then by definition, for any trajectory $\gamma \in T$ and any $r > 0$, there exist 2^D trajectories $T' \subset T$ such that any trajectory

$\gamma' \in \mathcal{B}_{\mathbf{d}_F}(\gamma, r)$ is at most at distance $\frac{r}{2}$ from a trajectory in T' . By the pigeon hole principle, any set of $2^D + 1$ or more trajectories in $\mathcal{B}_{\mathbf{d}_F}(\gamma, r)$ will have at least two trajectories that have the same closest trajectory in T' , and are therefore at most r distance apart (by the triangle inequality).

Consider a trajectory γ'' consisting of a sequence of points $\langle p_1, p_2, \dots, p_m \rangle$ in a straight line in \mathbb{R}^d , with a distance of at least $3r$ between every consecutive pair of points, for some $m, r > 0$. For each p_i , let $P_i = \{a_i, b_i\}$ be such that $\|a_i - b_i\| > r$ and $\|a_i - p_i\|, \|b_i - p_i\| \leq r$ (e.g., a_i and b_i can be the diametrically opposite points of the Euclidean ball of radius r centered at p_i). Consider the set of trajectories $T'' = \{\langle q_1, q_2, \dots, q_m \rangle \mid q_i \in P_i\}$. Then by construction, $T'' \subset \mathcal{B}_{\mathbf{d}_F}(\gamma'', r)$, and $\mathbf{d}_F(\gamma_1, \gamma_2) > r$ for any $\gamma_1, \gamma_2 \in T''$. Further, $|T''| = 2^m \geq 2^D + 1$ for sufficiently large m , contradicting the fact that $\mathcal{T} = (T, \mathbf{d}_F)$ has doubling dimension D .

2.7 Conclusion

We have given a framework for clustering where the cluster centers are restricted to belong to a simpler metric space. This is a reasonable definition for working with objects such as trajectories, where we may expect the data to contain noise due to the application domain. We characterized general conditions on this simpler space that allow us to obtain efficient $(1 + \varepsilon)$ -approximation algorithms for the k -median problem by proving we can use sampling to help solve the approximate 1-median problem. As special cases, we gave efficient algorithms for clustering trajectories and point sets under the discrete Fréchet and Hausdorff distances respectively.

We believe the general framework can be extended to other metric spaces as well. The next step would be to see if it can be applied to the continuous Fréchet distance, and to non-metric distance measures such as dynamic time warping. It would be interesting to provide other characterizations on the metric space for the cluster centers (as alternatives to the notion of covering discussed in this chapter) that are amenable to efficient clustering

algorithms.

Chapter 3

Locally Fair Partitioning

3.1 Introduction

Redistricting is a common societal decision making problem. In its basic form, there are two parties, say red and blue, and a parliament with some k representatives. Each individual (or voter) in the geographic region is aligned with one of the two parties. The goal is to divide the region into k parts – called districts – so that each part elects one representative to the parliament. It is typically assumed that each district does majority voting, so that if a district has more red voters than blue voters, then the chosen representative will be red.

The societal question then is *how should these districts be drawn?* One natural constraint is that the district is a connected region and, more preferably, has a compact shape. Another consideration is that each district is population *balanced*, i.e., has roughly the same number of individuals.¹ A final consideration, and one that will be the focus of this work, is *fairness*. If society has a large fraction of blue voters, the districts should not be drawn so that most representatives end up being red.

In this chapter, we consider a *local* and strong notion of fairness. Let us say that a voter is unhappy if she is in a majority blue (resp. red) district, but her party is red (resp. blue). We say that a given set of districts is *locally fair* if no subset of unhappy voters of the same party can deviate and form a feasible district (nicely shaped and balanced) so that they are the majority in that district. In other words, these deviating voters have a *justified complaint* – there was a different hypothetical district where they could have been happy. This notion is akin to that of the *core* from cooperative game theory [Sca67]. As such, if a partition is locally fair, then it is *as fair as possible* to the relevant parties – there are no

¹US courts have ruled that districts be population balanced, compact, and contiguous; see, e.g., https://en.wikisource.org/wiki/Reynolds_v._Sims.

groups could potentially form a region and do better.

There are examples in which a group of voters have argued they have a justified complaint regarding the redistricting. In the 2012 election in North Carolina, 13 House seats were allocated, 4 to Democrats and 9 to Republicans. In contrast, the percentage of voters who voted for a Democrat candidate was 50.60%. The U.S. Court of Appeals ruled that two of the districts’ boundaries in this map were unconstitutional due to gerrymandering and required new maps to be drawn. Considering this map, it is clear there exist compact potential districts which could be considered a deviating group with respect to the districting. This case (Cooper v. Harris (2017)) is just one in a long line of judgements on the *fairness* of districting plans in the U.S. ² The exhibition of deviating groups may help a political group or group of voters justify their complaint that a redistricting is unfair, and it may be effectively used in auditing proposed plans.

In contrast, some input instances may exhibit “natural gerrymandering”, when the distribution of the population prevents redistricting plans from being representative to all groups [BLSS18]. For example, if the minority party had 40% of the vote in total but the voters are uniformly distributed, it is unlikely that any deviating groups with a justified complaint would exist. In this case, one could argue no reasonable redistricting could ensure the minority group elects its fair share of the representatives. Thus, the notion of local fairness introduced in this work allows us to distinguish between natural and artificial gerrymandering. In contrast, when a redistricting plan is not *globally* fair (proportionally representative), it is not clear whether any group has a justified complaint regarding the redistricting, or if it is an unavoidable consequence of the geometry of the map.

3.1.1 Our Results

In this work, we study the existence and computation of a locally fair partition in the *one-dimensional case*, where we assume the n voters lie on a line or a circle. A *feasible*

²See also, Benisek v. Lamone (2018), Gill v. Whitford (2018), Rucho v. Common Cause (2019), etc.

district or region is now an interval containing $\sigma = n/k$ voters. The “niceness” aspect is captured by the region being an interval, and the “balance” aspect is captured by the number of points in each interval being n/k . Even in this setting, we show that locally fair partitioning is surprisingly non-trivial, and leads to a rich space of algorithmic questions.

Relaxed local fairness notions. In the 1D case, regulating each interval to be containing exactly σ voters is extremely restrictive, and it is relatively easy to show that even a balanced (not necessarily fair) partition need not exist. We will therefore allow ourselves to relax the interval size. We parameterize this by ε , so that the number of voters in any allowable interval lies in $[(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$. This also relaxes the number of intervals to be some number in $[\frac{k}{1+\varepsilon}, \frac{k}{1-\varepsilon}]$.³ Further, we also relax the notion of *deviation*, so that if a subset of voters deviate, they need to become “really happy” – they need to be a strict majority in the interval to which they deviate. We call this parameter $\beta \in [1/2, 1]$, so that unhappy points only deviate to a new allowable interval if their population size is at least $\beta\sigma$. If a fair partition exists under such relaxations, we term it (ε, β) -*fair*.

Under these relaxations, our first set of results in Section 3.3 characterizes the (ε, β) values for which a fair partition exists, and those where it may not. For $\varepsilon \leq 1/5$, we show a *sharp threshold* at $\beta = 1 - \varepsilon$: When $\beta < 1 - \varepsilon$, for large enough values of n , there is an instance with no (ε, β) -fair partition, while when $\beta \geq 1 - \varepsilon$, the simple strategy of creating uniform intervals is (ε, β) -fair. If we restrict points to deviate only when the interval they create has exactly σ points, this sharp threshold holds for all $\varepsilon \leq 1/3$. To interpret this result, when $\varepsilon = 1/3$, this means there is a fair partition where all intervals have size in the range $[\frac{2}{3}\sigma, \frac{4}{3}\sigma]$, and no subset of unhappy points can create an interval with σ points, where they form 2/3-majority. Furthermore, there is an instance where the bound of 2/3 on the majority cannot be reduced any further.

Beyond worst-case. The negative results above are adversarial: they need careful constructions of sequences of runs of red and blue points with precise lengths, so that

³Many of our results extend to the setting in which the number of intervals must be exactly k .

any partitioning scheme that needs intervals of certain size to eventually straddle both red and blue points in a way that allows a deviating interval to take shape. However, this is an artifact of the intervals needing almost precise balance, i.e., their lengths being approximately σ . The next question we ask is: suppose we are allowed to place a small fraction α of points in intervals whose sizes can be smaller than $(1 - \varepsilon)\sigma$. In particular, we could construct intervals for these points so that they are all happy, preventing them from deviating; or we could think of it as eliminating these points. Then *is it possible to circumvent these lower bounds?*

In Section 3.4, we show that the above is indeed the case when the input sequences are reasonably benign. By “benign”, we mean that the input is clustered, i.e., composed of runs of red and blue points of arbitrary lengths, as long as these lengths are lower bounded by some value ℓ . This models phenomena like *Schelling segregation* [Sch71, Zha11, IKLZ17], where individuals have a slight preference for like-minded neighbors and relocate to meet this constraint, which leads to “runs” of like-minded individuals.

For such input sequences, we show that as long as all runs are of length at least $\ell = 2\sigma$, once we allow a small fraction $\alpha = O(\frac{1}{k})$ of the points to be placed in unbalanced regions, there is a locally fair partition even for the strictest setting $(\varepsilon, \beta) = (0, 1/2)$: the remaining points are placed in intervals of size exactly σ , and no deviating interval of size σ has a simple majority of unhappy points.

Efficient partitioning. In Section 3.5, we finally study the algorithmic question: given parameters (ε, β) , decide whether a given input of length n admits a (ε, β) -fair partition. Note that the results so far have been worst-case existential results, and it is possible that even when $\beta < 1 - \varepsilon$, many inputs would have an (ε, β) -fair partition. The challenge in designing an algorithm is that a deviating interval could involve points from more than one interval in the partition. We resolve this via a dynamic programming algorithm whose running time is polynomial in n for any $\varepsilon \in [0, 1/2]$.

3.1.2 Related Work

Fairness notions. Proportionality is a classic approach to achieving fairness in social choice. In a proportional solution, different demographic slices of voters feel they have been represented fairly. This general idea dates back more than a century [Dro81], and has recently received significant attention [CC83, Mon95, BKS07, ABC⁺17, SFEL⁺17, AEH⁺18]. In fact, there are several elections, both at a group level and a national level, that attempt to find committees (or parliaments) that provide approximately proportional representation.

The notion of core from cooperative game theory [Sca67] represents the ultimate form of proportionality: every demographic slice of voters feel that they have been fairly represented and do not have incentive to deviate and choose their own solution which gives all of them higher utility. In the typical setting where these demographic slices are not known upfront, the notion of core attempts to be fair to all subsets of voters. Though the core has been traditionally considered in the context of resource allocation problems [Lin58, GS62, Fol70, SS74], one of our main contributions is to adapt this notion in a non-trivial way to the redistricting problem.

Redistricting vs. clustering. The redistricting problem is closely related to the clustering problem. In a line of recent work, various models of fairness have been proposed for center-based clustering. One popular approach to fairness ensures that each cluster contains groups in (roughly) the same proportion in which they exist in the population [CKLV17, ZVGRG17]. The redistricting problem we consider may take the opposite view – we effectively want the regions or clusters to be as close to monochromatic as possible to minimize the number of unhappy points in each region.

Chen et al. studied a variant of fair clustering problem where any large enough group of points with respect to the number of clusters are entitled to their own cluster center, if it is closer in distance to all of them [CFLM19]. This extends the notion of the core in a natural way to clustering. However, this work defines happiness of a point in terms of its distance, while in the redistricting problem, the happiness is in terms of the color of

the majority within that region. The latter leads to fundamentally different algorithmic questions.

Redistricting Algorithms. There has been extensive work on redistricting algorithms, going back to 1960s [HWS⁺65], for constructing contiguous, compact, and balanced districts. Many different approaches, including integer programming [God14], simulated annealing [AM10], evolutionary algorithms [LCW16a], Voronoi diagram based methods [SBD07, CAKY18], MCMC methods [BGH⁺17, DDS21a], have been proposed; see [BS20] for a recent survey. A line of work on redistricting algorithms focuses on combating manipulation such as gerrymandering: when district plans have been engineered to provide advantage to individual candidates or to parties [BLSS18]. For example, Cohen-Addad et al. propose a districting strategy with desirable geometric properties such as each district being a convex polygon with a small number of sides on average [CAKY18]. Using similar methods, Wheeler and Klein argue that the political advantage of urban or rural voters tends to be dramatically less than that afforded by district plans used in the real world [WK20]. In fact, Chen et al. show that district plans can also have unintentional bias arising from differences in geographic distribution of two parties [CR13].

Auditing. Another line of work in redistricting focuses on developing statistical tools to detect gerrymandering given a districting plan [HKL⁺20]. In many redistricting algorithms, existing methods generate maps without explicitly incorporating notions of fairness, but instead focusing on compactness. Popular methods generate an ensemble of plans and compare the number of representatives each party gets in the generated maps with the number received under the actual proposed maps [BS20]. In practice, political groups use many justifications for whether a plan is *fair*, and our work offers a new formal model which may be used for auditing—arguing that various plans satisfy properties of fairness [PTF22, DDS21a]. Our work in contrast takes a more algorithmic approach – given a natural definition of what a fair redistricting should look like, we show existence and computational results.

3.2 Preliminaries

Let X be a set of n points in \mathbb{R}^1 , each colored red or blue, and let $\sigma \in [n]$ be a parameter called ideal population size.⁴ We wish to construct a locally fair partition of X into intervals so that all intervals have roughly σ points. Only ordering of points in X really matters, so we describe the input as a (binary) sequence $X = x_1, \dots, x_n$, where $x_i \in \{\mathbf{R}, \mathbf{B}\}$ represents the color of the i -th point on the line.

Define $R := \{i \in [n] \mid x_i = \mathbf{R}\}$ and $B := \{i \in [n] \mid x_i = \mathbf{B}\}$ to be the subset of all red points and blue points, respectively. An *interval* is a contiguous sequence, defined by a pair of integers $i, j \in [n]$ and denoted as either $[i, j]$ (where both points i and j are included) or $(i, j]$ (where only j is included). For an interval $I \subset [n]$, let $|I|$ denote its size, i.e., $|[i, j]| = j - i + 1$, $|(i, j]| = j - i$.

An alternative way to describe the input. Sometimes it is useful to re-describe the input as a series of alternating *maximal* monochromatic intervals. When appropriate, we denote the input as $X = R_1, B_1, R_2, \dots, R_\eta, B_\eta$, where each $R_j \subseteq R$ (resp. $B_j \subseteq B$) is a maximal sequence of red (resp. blue) points, $R_j \neq \emptyset$ for $j > 1$, and $B \neq \emptyset$ for $j < \eta$. For each R_j, B_j , it suffices to specify its size.

Balanced Partition. We are interested in partitioning $[n]$ into pairwise-disjoint intervals, i.e., computing a partition $\Pi = \langle \pi_1, \dots, \pi_T \rangle$, where $T = |\Pi|$, $\pi_t = (i_{t-1}, i_t]$ for all $t \in [T]$, and $0 = i_0 < i_1 < \dots < i_T = n$.

We parameterize the population deviation in an interval using an input parameter ε : For $\varepsilon \in [0, 1/2]$, an interval $\pi_t \in \Pi$ is called ε -**allowable** (or allowable for brevity) if it satisfies $(1 - \varepsilon)\sigma \leq |\pi_t| \leq (1 + \varepsilon)\sigma$. The partition Π is **balanced** if each of its interval is allowable. Note that a balanced partition may not always exist (take $n = 100$, $\varepsilon = .01$, and $\sigma = 40$). In the remainder of the chapter, we assume that σ is chosen such that a balanced partition exists.

⁴We assume points lie on a line for simplicity; our results extend to the case of a ring in a straightforward manner.

For $\varepsilon = 0$, each interval has exactly σ points; and for $\varepsilon = 1/2$, each interval contains between $\frac{\sigma}{2}$ and $\frac{3\sigma}{2}$ points. In principle, we can choose ε to be any value in $[0, 1]$; but as the value of ε increases, the sizes of intervals in the partition become increasingly unbalanced. At an extreme, when $\varepsilon = 1$, every point could form its allowable interval. Thus, we only consider the setting of $\varepsilon \in [0, 1/2]$, though most of our results extend to settings of larger ε .

For an interval $I \subseteq [n]$, it is sometimes more convenient to work with the *ratio* $|I|/\sigma$, which is 1 for an interval of σ points. We define the *measure* of I , denoted by $\|I\|$, as $\|I\| = |I|/\sigma$.

Locally fair partition. Next, we turn our focus to defining the notion of local fairness. For an interval I , let $\mathcal{C}(I)$ represent its majority color. Formally, $\mathcal{C}(I) = \mathbf{R}$ if the interval I has red majority, i.e. $|R \cap I| > |B \cap I|$. Similarly, set $\mathcal{C}(I) = \mathbf{B}$ if $|R \cap I| < |B \cap I|$. Without loss of generality, if $|R \cap I| = |B \cap I|$, i.e., there is no majority in I , we define $\mathcal{C}(I) = \mathbf{B}$.

A point i is *happy* in Π if it is assigned to an interval matching its color, i.e., $\mathcal{C}(\pi_t) = x_i$, where $\pi_t \in \Pi$ is the interval containing i ; otherwise i is *unhappy* in Π . For an interval $I \subseteq [n]$, let $\text{uhp}(I, \Pi) := \{i \in I \mid i \text{ is unhappy in } \Pi\}$ denote the subset of unhappy points in I in partition Π . Note that here I can be any interval and is not necessarily a part in Π . If Π is fixed or clear from the context, we write $\text{uhp}(I) := \text{uhp}(I, \Pi)$.

Intuitively, a partition Π is *locally fair* if there is no large set of unhappy points which could form an allowable interval in which they would be the majority. We make this concept more precise below:

Definition 20. *Given an input instance (X, σ) and a parameter $\beta \in [\frac{1}{2}, 1]$, a **β -deviating group** with respect to a partition Π is an allowable interval D with more than $\max\left\{\frac{|D|}{2}, \beta\sigma\right\}$ unhappy monochromatic points, or equivalently,*

$$\max\{\|\text{uhp}(D) \cap R\|, \|\text{uhp}(D) \cap B\|\} > \max\left\{\frac{\|D\|}{2}, \beta\right\}.$$

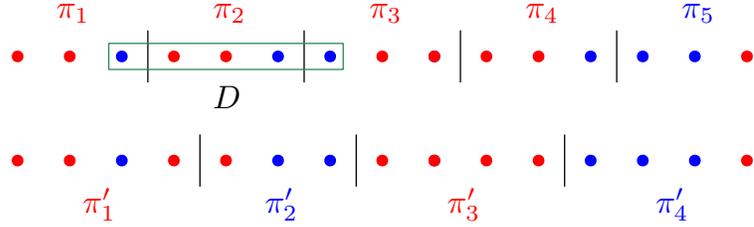


Figure 3.1: An instance with $n = 15$ points, $\sigma = 4$, $\varepsilon = 1/2$, and $\beta = 2/3$. The top partition $\{\pi_1, \dots, \pi_5\}$ admits a blue deviating group D , whereas the bottom partition $\{\pi'_1, \dots, \pi'_4\}$ is $(1/2, 2/3)$ -locally fair.

Note that we do not require the existence of a balanced partition with D being one of its intervals.

D is called a β -deviating group because D may deviate from Π such that at least $\beta\sigma$ points that were unhappy in Π become happy if D was made a standalone part in another partition. We sometimes omit β and use the term “deviating group” when the context is clear. Intuitively, β controls how difficult it is for a set of unhappy points to deviate and form an interval in which they are happy. As β grows, the set of unhappy points that may form a deviating group must grow larger with respect to the desired interval size σ ; to deviate when $\beta = 1/2$, a set of more than $\sigma/2$ unhappy points must lie in an allowable interval in which they are the majority. At $\beta = 1$, the number of unhappy points required to form a deviating group increases to σ . See Figure 3.1 for an example.

Definition 21. *Given (X, σ) , $\varepsilon \in [0, 1/2]$, and $\beta \in [1/2, 1]$, we call a balanced partition Π (ε, β) -locally fair if there is no β -deviating group with respect to Π .*

Remark. While we focus on $\beta \in [1/2, 1]$, the largest possible range to consider is $\beta \in [(1 - \varepsilon)/2, (1 + \varepsilon)]$. For applications of our model, we would expect the requirement on β to be stronger than a simple majority, so that we bias towards solutions (partitions) which are presumably fair to the rest of the points.

3.3 Existence of Locally Fair Partitions

In this section, we present our results on the existence of locally fair partitions. We first give characterizations of parameters ε and β for which a locally fair solution is guaranteed to exist. We show that for every $\varepsilon \in [0, 1/2]$, there is a threshold $\bar{\beta}(\varepsilon)$ such that if β is above this threshold, a simple partitioning strategy into small intervals results in a (ε, β) -fair partition.

Theorem 22. *Let (X, σ) be an input instance as defined above. For any $\varepsilon \in [0, 1/2]$, there is a value $\bar{\beta}(\varepsilon)$ such that for any $\beta > \bar{\beta}(\varepsilon)$, there exists an (ε, β) -fair partition, where*

$$\bar{\beta}(\varepsilon) = \begin{cases} \max \left\{ 1 - \varepsilon, \frac{1+3\varepsilon}{2} \right\} + O(\delta) & \text{for } \varepsilon \in [0, \frac{1}{3}], \\ \max \left\{ \frac{3(1-\varepsilon)}{2}, 2\varepsilon \right\} + O(\delta) & \text{for } \varepsilon \in [\frac{1}{3}, \frac{1}{2}], \end{cases}$$

and $\delta \leq \frac{1}{\frac{n}{\sigma}-1} + \frac{1}{\sigma}$.

Proof. We first consider the case of $\varepsilon \in [0, 1/3]$. We call a partition Π of $[n]$ *almost-uniform* if there is some $q \in [1 - \varepsilon, 1 + \varepsilon]$ such that $\sigma' = q\sigma$ is an integer, and $|\pi_t| \in \{\sigma' - 1, \sigma'\}$ for all $t = 1, \dots, |\Pi|$. Consider the following partition: construct $\lfloor \frac{n}{(1-\varepsilon)\sigma} \rfloor$ intervals of $(1-\varepsilon)\sigma$ points each. Then, uniformly distribute the remaining points (less than $(1-\varepsilon)\sigma$ of them) across the intervals. Every interval then gets at most $\left\lceil \frac{(1-\varepsilon)\sigma}{\lfloor \frac{n}{(1-\varepsilon)\sigma} \rfloor} \right\rceil$ additional points. Therefore, the total number of points in the largest interval is given by

$$\begin{aligned} \sigma' = q\sigma &= (1-\varepsilon)\sigma + \left\lceil \frac{(1-\varepsilon)\sigma}{\lfloor \frac{n}{(1-\varepsilon)\sigma} \rfloor} \right\rceil \\ &\leq (1-\varepsilon)\sigma + \frac{(1-\varepsilon)\sigma}{\frac{n}{(1-\varepsilon)\sigma} - 1} + 1 \\ &\leq (1-\varepsilon)\sigma + \frac{(1-\varepsilon)^2\sigma^2}{n - (1-\varepsilon)\sigma} + 1. \end{aligned}$$

Hence we have

$$\delta = q - (1-\varepsilon) \leq \frac{(1-\varepsilon)^2\sigma}{n - (1-\varepsilon)\sigma} + \frac{1}{\sigma} \leq \frac{\sigma}{n - \sigma} + \frac{1}{\sigma} = \frac{1}{\frac{n}{\sigma} - 1} + \frac{1}{\sigma},$$

when $n \gg \sigma$.

Hence take the almost-uniform partition Π with this value of q . Consider any potential deviating group D of Π . First, suppose D intersects at least four contiguous intervals of Π . Then it must contain at least two intervals in the middle, and therefore $|D| > 2(1 - \varepsilon)\sigma \geq (1 + \varepsilon)\sigma$ for $\varepsilon \in [0, 1/3]$, which violates the requirement that D must be allowable. Hence, D can intersect at most three (contiguous) intervals of Π . If D intersects one or two intervals of Π , then

$$|\text{uhp}(D, \Pi)| \leq 2 \cdot \frac{\sigma'}{2} \leq \sigma' = q\sigma \leq \bar{\beta}(\varepsilon)\sigma < \beta\sigma,$$

which implies D cannot be a deviating group itself. Hence, no deviating group intersects at most 2 intervals. Finally, if D intersects 3 intervals, $\{\pi_t, \pi_{t+1}, \pi_{t+2}\}$, then it completely contains the middle interval π_{t+1} , which has size at most σ' . Since at most half of the points in π_{t+1} are unhappy, we have $|\text{uhp}(\pi_{t+1})| \leq \frac{\sigma'}{2}$. On the other hand, we have

$$|D \cap (\pi_t \cup \pi_{t+2})| \leq (1 + \varepsilon)\sigma - |\pi_{t+1}| \leq (1 + \varepsilon)\sigma - (\sigma' - 1).$$

Thus we have

$$\begin{aligned} |\text{uhp}(D, \Pi)| &= \sum_{j=t}^{t+2} |\text{uhp}(D \cap \pi_j)| \leq |D \cap (\pi_t \cup \pi_{t+2})| + |\text{uhp}(\pi_{t+1})| \\ &\leq (1 + \varepsilon)\sigma - \sigma' + 1 + \frac{\sigma'}{2} \\ &\leq \left(1 + \varepsilon - \frac{q}{2} + \frac{1}{\sigma}\right)\sigma \\ &\leq \left(\frac{1 + 3\varepsilon}{2} - \frac{\delta}{2} + \frac{1}{\sigma}\right)\sigma \\ &\leq \bar{\beta}(\varepsilon)\sigma < \beta\sigma, \end{aligned}$$

which again implies D cannot be a deviating group. Hence D does not exist, and the proposed almost-uniform partition Π is (ε, β) -locally fair for $\varepsilon \in [0, 1/3]$. The proof for $\varepsilon \in [1/3, 1/2]$ follows analogously by increasing the largest number of intervals the deviating group can intersect by one, from three to four. \square

Remark. When $n = c(1 - \varepsilon)\sigma$ for some integer c , the partition proposed in the proof of Theorem 22 becomes a uniform partition with interval size $(1 - \varepsilon)\sigma$, and thus $\delta = 0$.

Otherwise, we try to create an almost-uniform partition such that the intervals are as small as possible. For Theorem 22 to give a $\bar{\beta}(\varepsilon)$ bound below 1, we need δ to be roughly $O(\varepsilon)$, or $\varepsilon > \sigma/n$. If $\sigma/n = c$ for a constant c , the desired interval size is $\Theta(n)$, and the number of intervals in a balanced partition will only be $O(1)$. In this setting, it is likely that any fair partitioning scheme must take into account the coloring of the input.

Theorem 22 can be extended for any $\varepsilon \in [0, 1]$. Following the same proof approach gives the general form of $\bar{\beta}(\varepsilon) = \max \left\{ \frac{t(1-\varepsilon)}{2}, \frac{(3-t)+(t+1)\varepsilon}{2} \right\} + O(t\sigma)$, where t is an integer such that $\varepsilon \in \left[\frac{t-2}{t}, \frac{t-1}{t+1} \right]$, or in other words, $(t+1)$ is the largest number of intervals a deviating group can intersect.

Next, we show that for smaller values of β , a locally fair partition may not exist.

Theorem 23. *Let $\varepsilon \in [0, 1/2)$ and $\beta \in [1/2, 1 - \varepsilon)$. For any $\sigma \geq 1$, there exists an input instance (X, σ) with $|X| = O\left(\frac{\beta\sigma}{1-\varepsilon-\beta}\right)$ for which no (ε, β) -locally fair partition exists.*

Proof. We construct an instance for which there is always a deviating group. For simplicity, assume $\beta\sigma$ is an integer; we will relax this assumption later. Specifying the input using the runs of monochromatic intervals, let $X = R_1, B_1, R_2, B_2, \dots, R_\eta, B_\eta$ for $\eta = \left\lceil \frac{n}{2\beta\sigma} \right\rceil$. Set $|R_j| = \beta\sigma$ for $j = 1, \dots, \eta$ and $|B_j| = \beta\sigma$ for $j = 1, \dots, \eta-1$, and $|B_\eta| = n - (2\eta-1)\beta\sigma \leq \beta\sigma$.

In any fair partition Π , each R_j (resp. B_j) must intersect a red (resp. blue) interval of Π ; if there exists an entire R_j (resp. B_j) contained in a blue (resp. red) interval of Π , R_j (resp. B_j) could form a deviating group with its own $\beta\sigma$ unhappy points, and $(1 - \varepsilon - \beta)\sigma$ points from a neighboring B_j (resp. R_j). Since $\beta \geq 1/2$, these unhappy red (resp. blue) points are the majority in the deviating group. As a consequence, in any fair partition Π , there exists no red (resp. blue) interval π_t that intersects multiple monochromatic red intervals $R_j, R_{j'}$ (resp. blue intervals $B_j, B_{j'}$) in X .

Suppose there exists a fair partition Π for (X, σ) , and let π_1 be a red interval of Π that intersects R_1 . Since $\beta < 1 - \varepsilon$, π_1 must include points from a neighboring blue monochromatic interval; without loss of generality, let it intersect with B_1 .⁵ Then π_1 cannot

⁵We assume the input lies on a circle. Since π_1 must intersect at least one of the two blue

intersect R_2 ; otherwise B_1 would deviate. Therefore, π_1 includes at most $\beta\sigma$ points from R_1 and some points from B_1 . Now, consider interval π_2 of Π which is blue and intersects B_1 (but not B_2). By size constraints, $\|\pi_2\| \geq (1 - \varepsilon)$, and $\|\pi_2 \cap B_1\| < \beta$, since $\pi_1 \cap B_1 \neq \emptyset$. This implies $\|\pi_2 \cap R_2\| > (1 - \varepsilon - \beta)$. Next, interval π_3 is red-majority and intersects R_2 (but not R_3); moreover, we have $\|\pi_3 \cap R_2\| \leq \|R_2\| - \|\pi_2 \cap R_2\| < \beta - (1 - \varepsilon - \beta)$. This then implies $\|\pi_3 \cap B_2\| \geq \|\pi_3\| - \|\pi_3 \cap R_2\| > (1 - \varepsilon) - (\beta - (1 - \varepsilon - \beta)) = 2(1 - \varepsilon - \beta)$.

Continuing this argument, it can be shown that (i) every R_j must intersect a red interval in Π that also intersects with B_j ; and (ii) every B_j must intersect a blue interval in Π that also intersects with R_{j+1} . Combined with the fact that each red- (resp. blue-) majority π_t cannot intersect multiple monochromatic red (resp. blue) intervals of X , for every j it must hold that:

- π_{2j-1} is red-majority and intersects R_j ;
- π_{2j} is blue-majority and intersects B_j ;
- $\|\pi_{2j-1} \cap R_j\| < \beta - (2j - 3) \cdot (1 - \varepsilon - \beta)$;
- $\|\pi_{2j-1} \cap B_j\| > (2j - 2) \cdot (1 - \varepsilon - \beta)$;
- $\|\pi_{2j} \cap B_j\| < \beta - (2j - 2) \cdot (1 - \varepsilon - \beta)$;
- $\|\pi_{2j} \cap R_{j+1}\| > (2j - 1) \cdot (1 - \varepsilon - \beta)$.

Denote $j' = \left\lceil \frac{3-3\varepsilon-2\beta}{4(1-\varepsilon-\beta)} \right\rceil$, and assume $\eta \geq j'$. By the above, $\pi_{2j'}$ is blue-majority, and we have $\|\pi_{2j'} \cap B_{j'}\| < \beta - (2j' - 2) \cdot (1 - \varepsilon - \beta) < \frac{1-\varepsilon}{2}$, which implies $\pi_{2j'}$ cannot be blue-majority, a contradiction. In other words, there are not enough points in $\pi_{2j'}$ to create a majority matching its color. Since $\eta = \left\lceil \frac{n}{2\beta\sigma} \right\rceil$, the above holds for $\frac{n}{2\beta\sigma} > \frac{3-3\varepsilon-2\beta}{4(1-\varepsilon-\beta)}$, or $n > \frac{(3-3\varepsilon-2\beta)\beta\sigma}{2(1-\varepsilon-\beta)} = O\left(\frac{\beta\sigma}{1-\varepsilon-\beta}\right)$. Finally, if $\beta\sigma$ is not an integer, let $\beta' = \frac{\lceil \beta\sigma \rceil}{\sigma}$. Then the above argument still holds for $n > \frac{(3-3\varepsilon-2\beta')\beta'\sigma}{2(1-\varepsilon-\beta')}$. \square

monochromatic intervals neighboring R_1 , we can order the input so that the intersected interval is B_1 .

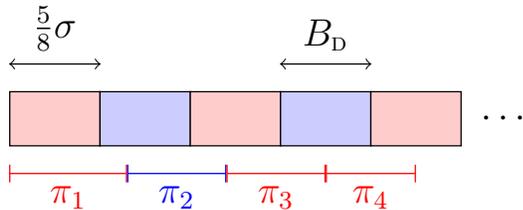


Figure 3.2: An instance so that each monochromatic interval has length $5\sigma/8$, which does not admit a $(\frac{1}{4}, \frac{5}{8})$ -fair partition. For example, partition Π is made of intervals of length $(1 - \varepsilon)\sigma = \frac{3\sigma}{4}$; however, B_D forms a deviating group by pulling in points from neighboring intervals.

See Figure 3.2 for an example of the construction. For $\varepsilon \in [0, 1/5]$, Theorem 22 and Theorem 23 provide an almost sharp threshold: if $\beta > 1 - \varepsilon + O(\delta)$ (for δ defined in Theorem 22), a locally fair partition always exists, but for $\beta < 1 - \varepsilon$ there are instances that do not admit fair partitions. In fact, if we enforce the deviating group to have exactly σ points, Theorems 22 and 23 become almost tight for all $\varepsilon \in [0, 1/3]$. We next extend Theorem 23 so that a single instance X has no locally fair partition for a wide range of σ values.

Corollary 24. *Let $\varepsilon \in [0, 1/2)$, $\beta \in [1/2, 1 - \varepsilon)$, and let $S = \{\sigma_1, \sigma_2, \dots, \sigma_M\}$ be the set of desired interval sizes. If $\frac{n}{M\sigma_m} > \left\lceil \frac{1}{1 - \varepsilon - \beta} \right\rceil$ holds for all $m \in [M]$, there exists an input X such that (i) $|X| = n$; (ii) for all $m = 1, \dots, M$, the instance (X, σ_m) has no (ε, β) -locally fair partition.*

Proof. We construct X as follows. Let $X_1, \dots, X_M \subseteq X$ be subintervals of X , each of size n/M . For each part X_m , we apply Theorem 23 with (X_m, σ_m) . For each σ_m , a deviating group exists in X_m . □

Remark. (i) In light of Theorem 23, we observe that if we are given a periodic instance, it is always possible to define a $\tilde{\sigma}$ for which the instance has a locally fair solution, even if the instance had no locally fair solution for the given σ . Specifically, $\tilde{\sigma} = \Theta(\sigma)$. In the next section, we define an approximate version of the problem, in which we allow some fraction of points to lie outside of allowable regions. We focus on “periodic-like” instances, when

the input consists of monochromatic intervals of size $\Omega(\sigma)$.

(ii) In some applications, it may be desirable to ensure there is a fixed number of intervals in a partition. Given a desired number of intervals k , the negative results of Theorem 23 extend to this setting, i.e., we can construct periodic instances similar to those in Theorem 23 in which no locally fair solutions exist, for even larger ranges of (ε, β) parameters.

3.4 Clustered Instances

As manifested in the previous section, under many specific parameters ε, β , there exist adversarial input instances (X, σ) that rule out the existence of any locally fair partition. However, such negative instances often seem artificial, and are not robust to perturbation. In this section, we turn our attention to a category of interesting inputs, when points are “clustered” into large monochromatic intervals. Such instances arise in applications in which we expect points of the same color to gather together. We show that fair partitions exist when the input instance is comprised of large monochromatic intervals, while incurring a small approximation on the balancedness of the fair partition.

For a constant $\alpha \in [0, 1)$, we say a partition Π of X is **α -balanced** if the union of all its allowable intervals make up at least a $(1 - \alpha)$ -fraction of the total input. Formally, let $\tilde{\Pi} := \{\pi_t \in \Pi \mid |\pi_t| \in [(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]\}$ be the set of allowable intervals in Π . Then Π is α -balanced if $\left| \bigcup_{\pi_t \in \tilde{\Pi}} \pi_t \right| \geq (1 - \alpha)n$.

In fact, in this section our results hold for any $\beta \in [1/2, 1]$, so we omit β as a parameter, and instead refer to a fair partition as ε -locally fair.

First, we show that if the size of each monochromatic interval in X is at least 2σ , we can compute a fair partition by letting allowable intervals not contain a small fraction of the population.

Theorem 25. *Given instance $(X = R_1, B_1, R_2, \dots, \sigma)$ with $\|R_j\|, \|B_j\| \geq 2$, for all j and parameter $\varepsilon \in [0, 1/2]$, there is a $\left(\frac{(1-\varepsilon)\sigma}{n}\right)$ -balanced, ε -locally fair partition Π .*

Proof. We assume that the first monochromatic interval R_1 is red and the last monochromatic interval B_η is blue. The other cases follow analogously. Consider an arbitrary maximal monochromatic red interval R_j with measure $\|R_j\| \geq 2$. We divide R_j into allowable intervals such that the residual interval R'_j (points of R_j not assigned to an allowable interval) is as small as possible. Note that $\|R'_j\| \in [0, 1 - \varepsilon)$. We assign the points of R'_j to a size $\lceil (1 - \varepsilon)\sigma \rceil$ interval using $\lceil ((1 - \varepsilon) - \|R'_j\|)\sigma \rceil$ points from the next interval, B_j . Partition the entire instance in this manner and let Π be the resulting partition. All intervals of Π are allowable except for the residual interval B'_η from the last monochromatic interval B_η . Since $|B'_\eta| < (1 - \varepsilon)\sigma$, Π is $\left(\frac{(1-\varepsilon)\sigma}{n}\right)$ -balanced.

We next show Π is locally fair regardless of the value of ε . Suppose there exists a deviating group D . Without loss of generality, assume D is red-majority and D intersects two consecutive red intervals R_j and R_{j+1} of X . Then we have $B_{j+1} \subset D$. Since $\|B_{j+1}\| \geq 2$, we have $\|D\| > 2 > (1 + \varepsilon)$, a violation of the size constraint. Hence, D can only intersect one monochromatic red interval R_j for some j .

Define $R''_j \subseteq R_j$ (resp. $R'_j \subseteq R_j$) to be the (not necessarily non-empty) set of red points assigned to the same interval, denoted by π'' (resp. π') as a subset B'_j of B_j (resp. B'_{j+1} of B_{j+1}) (See Figure 3.3). If π' is blue majority, then $\|R''_j\| < \frac{1-\varepsilon}{2}$, by construction. Similarly, if π'' is blue majority, then $\|R'_j\| < \frac{1-\varepsilon}{2}$. Assume D intersects both R'_j and R''_j . Then both π' and π'' need to be blue-majority, and we have $R \setminus (R'_j \cup R''_j) \subset D$. But this implies $\|R \setminus (R'_j \cup R''_j)\| > 2 - \frac{1-\varepsilon}{2} - \frac{1-\varepsilon}{2} = 1 + \varepsilon$ and thus $\|D\| > 1 + \varepsilon$, a contradiction.

Hence, D can only intersect either R'_j or R''_j . In both cases, $\|\text{uhp}(D)\| < \frac{1-\varepsilon}{2}$, implying D does not have sufficient points to form a deviating group. Therefore, no deviating group exists in Π . \square

In fact, we can use the same partitioning strategy to find balanced fair partitions (i.e., every point belongs to an allowable interval π_t) if each monochromatic interval of an instance has size at least $\left\lceil \frac{(1-\varepsilon)^2}{2\varepsilon} \right\rceil$.

Corollary 26. *For an instance (X, σ) and parameter ε , such that for all j : $\|R_j\|, \|B_j\| \geq \left\lceil \frac{(1-\varepsilon)^2}{2\varepsilon} \right\rceil$, there is always a balanced ε -locally fair partition Π of X .*

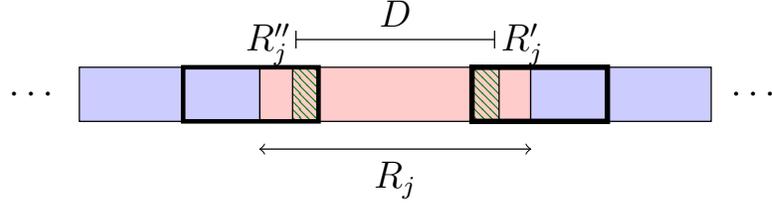


Figure 3.3: No deviating group D intersects R_j'' and R_j' .

Proof. Observe that the residual interval for each R_j, B_j defined in Theorem 25 has length zero. Accordingly, by uniformly partitioning each R_j, B_j into equally-sized monochromatic intervals, we have a locally fair partition of X in which no points are unhappy. \square

Remark. For all $\varepsilon \geq 3 - 2\sqrt{2} \approx .17$, Corollary 26 requires monochromatic intervals to have size at most that of Theorem 25, but achieves exact balancedness. However, the threshold $\left\lceil \frac{(1-\varepsilon)^2}{2\varepsilon} \right\rceil$ is non-increasing in ε : when $\varepsilon \rightarrow 0$, $\left\lceil \frac{(1-\varepsilon)^2}{2\varepsilon} \right\rceil \rightarrow \infty$, while when $\varepsilon \geq 1/2$, $\left\lceil \frac{(1-\varepsilon)^2}{2\varepsilon} \right\rceil = 1$. Thus, for small values of ε , it is preferable to apply Theorem 25.

Next, we relax the requirement that *every* monochromatic interval is long and consider “mostly” clustered instances. Specifically, assume that at most γn points of X lie in monochromatic intervals smaller than size 2σ . Applying Theorem 25 to this setting, we can construct an α -balanced fair partition with α depending on γ .

Theorem 27. *Given (X, σ) and parameter $\varepsilon \in [0, 1/2]$, where $X = R_1, B_1, \dots, R_\eta, B_\eta$, let Y denote the set of monochromatic intervals of X of length smaller than 2σ : $Y := \{I \in \{R_1, \dots, R_\eta, B_1, \dots, B_\eta\} \mid \|I\| < 2\sigma\}$.*

If $|\bigcup_{I \in Y} I| \leq \gamma n$, then there is a $\left(\frac{(1-\varepsilon)\sigma}{n} + \gamma\right)$ -balanced, ε -locally fair partition.

Proof. We partition X into two parts: the *bad* part Y and the *good* part $Z = X \setminus Y$. X can be regarded as an alternating sequence of Z_i s and Y_j s.

Apply Theorem 25 to each good part Z_i . For each Z_i , the residual points Z_i' of the last monochromatic interval in Z_i satisfies $\|Z_i'\| < (1 - \varepsilon)\sigma$. Then we consider two cases:

- $\|Z_i'\| + \|Y_i\| \geq 1 - \varepsilon$. We define an interval I_i containing Z_i' and $(1 - \varepsilon - \|Z_i'\|)\sigma$ points of Y_i , and add I_i to the partition. For each remaining monochromatic interval R_j (or

B_j) in Y_i , we create a (not necessarily allowable) standalone interval in our partition, so that all points are happy and do not contribute to any potential deviating groups.

- $\|Z'_i\| + \|Y_i\| < 1 - \varepsilon$. In this case, we apply the partitioning scheme in the proof of Theorem 25 on Z_i, Y_i, Z_{i+1} , namely, there is an interval π_i containing Z'_i, Y_i and $\lceil (1 - \varepsilon - |Z_i| - |Y_i|)\sigma \rceil$ points of the first monochromatic interval of Z_{i+1} .

In both cases, no deviating group exists since the interval containing Z'_i and points of Y_i are adjacent to intervals with no unhappy points, although there may be a total of γn points lying in non-allowable intervals $\pi_i \in \Pi$ with $|\pi_i| < (1 - \varepsilon)\sigma$. Additionally, for the last good part Z_i , there may be an additional $\left(\frac{(1-\varepsilon)\sigma}{n}\right)$ points lying in non-allowable intervals, as a direct consequence of applying Theorem 25. (Note that its counterparts in Z_1, \dots, Z_{i-1} are handled in the above process.) Thus, there is a $\left(\frac{(1-\varepsilon)\sigma}{n} + \gamma\right)$ -balanced, ε -locally fair partition. \square

For all ε , α improves (i.e., decreases) as γ decreases, as more of the input lies in larger monochromatic intervals. Similar to Corollary 26, if $\varepsilon \geq 3 - 2\sqrt{2}$, we can prove the above process gives a γ -balanced, ε -locally fair partition.

Remark. Here we defined the approximation concept in terms of the balancedness of fair partitions. If we want to ensure the partition is strictly balanced, we can instead define the approximation concept in terms of the total number of points in the union of deviating groups, i.e., we allow a maximum of α -fraction of the points to deviate (participate in some deviating group). Still assuming that at most γn points in the input lie in monochromatic intervals smaller than size 2σ , if $\varepsilon \geq 3 - 2\sqrt{2}$, we can get an exactly balanced partition which is $\left(\frac{\gamma}{2} + \frac{1-\varepsilon}{4}\right)$ -approximately fair. For smaller ε , we still require a non-zero α , but the γ parameter could be moved into the fairness approximation.

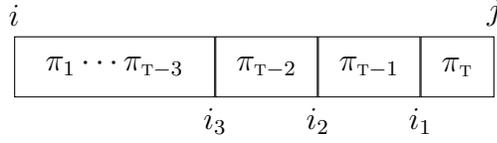


Figure 3.4: Π is a partition of interval $(i, j]$, where i_1, i_2 , and i_3 are the last three boundaries.

3.5 Partitioning Algorithm

Finally, we shift our focus to following algorithmic question: *Given an instance (X, σ) and parameters $\varepsilon \in [0, 1/2], \beta \in [1/2, 1]$, does a (ε, β) -locally fair solution exist for (X, σ) ?*

In this section, we focus on a fixed input instance, so throughout this section, treat X, σ, ε , and β as fixed, and describe an algorithm that determines whether an (ε, β) -locally fair balanced partition, possibly with additional constraints, exists, for an interval $I \subseteq [n]$ of the input, where $|X| = n$.

We now define the recursive subproblems. For any interval $I = (i, j] \in [n]$ and for $i \leq i_3 \leq i_2 \leq i_1 < j$, define $\text{LF}(I, i_1, i_2, i_3) = \text{True}$ if and only if there exists at least one fair partition $\Pi = \{\pi_1, \dots, \pi_T\}$ of I that satisfies the following conditions:

$$\begin{aligned}
 i_1 = i_2 = i_3 = i, \quad \Pi &= \{(i, j]\}, \quad \text{for } T = 1; \\
 i_1 > i_2 = i_3 = i, \quad \Pi &= \{(i, i_1], (i_1, j]\}, \quad \text{for } T = 2; \\
 i_1 > i_2 > i_3 = i, \quad \Pi &= \{(i, i_2], (i_2, i_1], (i_1, j]\}, \quad \text{for } T = 3; \\
 i_1 > i_2 > i_3 > i, \quad \Pi &= \{\pi_1, \dots, (i_3, i_2], (i_2, i_1], (i_1, j]\}, \text{ o/w.}
 \end{aligned}$$

In other words, i_1, i_2 , and i_3 are the last three “interval boundaries” in Π ; see Figure 3.4.

We first define the base cases of our algorithm. Consider every interval $I = (i, j]$ that is allowable, i.e., $(1 - \varepsilon)\sigma \leq |I| \leq (1 + \varepsilon)\sigma$. Without loss of generality, let $\chi(I) = \mathbf{B}$. We consider letting $\Pi = \{(i, j]\}$ be the trivial partition for I . This is locally fair if and only if no deviating group can form within I , i.e., there exists no $(i', j'] \subset (i, j]$ such that (i) $(1 - \varepsilon)\sigma \leq (j' - i')$ (so that $(i', j']$ is allowable), and (ii) $|\text{uhp}((i', j'] \cap R)| \geq \max\{\frac{\beta\sigma}{2}, \frac{j'-i'}{2}\}$ (so that $(i', j']$ is deviating). If the above holds, we have $\text{LF}(I, i, i, i) = \text{True}$, and $\text{LF}(I, i_1, i_2, i_3) = \text{False}$ for any other values of i_1, i_2 , and i_3 .

Intuitively, if an interval $I = (i, j] \subseteq [n]$ has a fair partition, either I is a standalone fair allowable interval (i.e., $\text{LF}((i, j], i, i, i) = \text{True}$), or there must exist a point $j' \in [j - (1 + \varepsilon)\sigma, j - (1 - \varepsilon)\sigma)$ such that

(i) there is a fair partition for $(i, j']$;

(ii) $(j', j]$ is a fair unpartitioned standalone interval;

(iii) no deviating group is formed by the last 3 intervals of fair partition of $(i, j']$ and the interval $(j', j]$.

To see this, simply observe that any fair partition of $(i, j]$ that contains at least two intervals has at least one interval boundary j' between $(j - (1 + \varepsilon)\sigma)$ and $(j - (1 - \varepsilon)\sigma)$ (so that the last interval $(j', j]$ is allowable). There is no deviating group formed within interval $(i, j']$, $(j', j]$ or straddling j' (such a deviating group cannot span more than three intervals of a balanced partition of $(i, j']$). Accordingly, we have the following lemma:

Lemma 28. $\text{LF}((i, j], i_1, i_2, i_3) = \text{True}$ if and only if there exists a point $i_4 \in [\max\{i, i_3 - (1 + \varepsilon)\sigma\}, i_1]$ such that

- $\text{LF}((i_1, j], i_1, i_1, i_1) = \text{True}$, and
- $\text{LF}((i, i_1], i_2, i_3, i_4) = \text{True}$, and
- There is no deviating group forming within $(i_4, j]$ with the partition

$$\{(i_4, i_3], (i_3, i_2], (i_2, i_1], (i_1, j]\}.$$

Proof. Recall that $\varepsilon \leq \frac{1}{2}$. Thus, every allowable interval, as well as a potential deviating group, has size between $\frac{\sigma}{2}$ and $\frac{3\sigma}{2}$. Suppose there exists a deviating group D for a partition Π of $(i, j]$ intersecting more than four contiguous intervals in Π . Then D must strictly contain at least three intervals $\pi_i, \pi_{i+1}, \pi_{i+2}$, and thus we have $|D| > 3 \cdot \frac{\sigma}{2}$, a contradiction. Hence D can intersect at most four contiguous intervals in any partition Π for X . Suppose $\text{LF}((i_1, j], i_1, i_1, i_1) = \text{LF}((i, i_1], i_2, i_3, i_4) = \text{True}$ for some i_4 . This means (i) interval $(i_1, j]$

is fair as a standalone interval; (ii) there is a fair partition Π' of interval $(i, i_1]$, with the last three interval boundaries being i_2 , i_3 , and i_4 . Concatenate Π' with $(i_1, j]$ as Π . By the discussion above, any deviating group D for Π intersects at most four contiguous intervals in Π . Suppose a such D intersects $(i, i_4]$. Then it must not intersect $(i_1, j]$ as a consequence, which implies D is also a deviating group for Π' , a contradiction. Hence D must be contained in $(i_4, j]$. However, by the third criteria, there is no deviating group in $(i_4, j]$ either. Therefore, no deviating group exists in D , which means Π is a fair partition for $(i, j]$. \square

In other words, for i_1, i_2 , and i_3 to be the last three interval boundaries in a fair partition for $(i, j]$, the last interval $(i_1, j]$ must be fair itself, and there must exist a point i_4 as the next (fourth last) interval boundary. Note that it is possible that $i_4 = i_3$ in the degenerate case when $(i, j]$ is partitioned into less than 4 intervals. Accordingly, we know that $(i, i_1]$ must also be fair with last three interval boundaries being i_2, i_3 , and i_4 , as well as $(i_4, j]$ must be fair with the interval boundaries being i_1, i_2 , and i_3 .

For $\varepsilon \leq 1/3$, as $(i_4, i_3]$ cannot be further separated into multiple allowable intervals, Lemma 28 can be simplified as follows:

Corollary 29. *For $\varepsilon \leq 1/3$, $\text{LF}((i, j], i_1, i_2, i_3) = \text{True}$ if and only if there exists a point $i_4 \in [\max\{i, i_3 - (1 + \varepsilon)\sigma\}, i_1]$ such that*

- $\text{LF}((i_1, j], i_1, i_1, i_1) = \text{True}$, and
- $\text{LF}((i, i_1], i_2, i_3, i_4) = \text{True}$, and
- $\text{LF}((i_4, j], i_1, i_2, i_3) = \text{True}$.

Proof. The proof follows from Lemma 28. Since $\text{LF}((i_4, j], i_1, i_2, i_3) = \text{True}$, there must exist a fair partition for $(i_4, j]$ with the last three intervals being $\{(i_3, i_2], (i_2, i_1], (i_1, j]\}$. By the fact that $\text{LF}((i, i_1], i_2, i_3, i_4) = \text{True}$, we know $(i_4, i_3]$ is an allowable region. For $\varepsilon \in [0, 1/3)$, this implies $|(i_4, i_3]| \leq (1 + \varepsilon)\sigma < \frac{2\sigma}{3} < \frac{2(1-\varepsilon)\sigma}{3}$, i.e., there is no other way to partition $(i_4, i_3]$ in a balanced fashion other than making it a standalone part. Hence,

the partition $\{(i_4, i_3], (i_3, i_2], (i_2, i_1], (i_1, j]\}$ must be the only fair partition for $(i_4, j]$ with the last three intervals being $\{(i_3, i_2], (i_2, i_1], (i_1, j]\}$, which implies the third criterion in Lemma 28 is met. \square

Hence, for a general interval $(i, j]$, to compute $\text{LF}((i, j], i_1, i_2, i_3)$, it suffices to check for all possible values of i_4 , each incurring two lookups of previously computed subquery results and one check of fairness in a partition for an interval of length at most $4(1 + \varepsilon)\sigma$. For $\varepsilon \in [0, 1/3)$, it can be simplified to three lookups of previously computed subquery results.

Algorithm. We use Corollary 29 and dynamic programming to compute

$$\bigvee_{1 \leq i_1 \leq i_2 \leq i_3 \leq n} \text{LF}([0, n], i_1, i_2, i_3),$$

as follows. Our algorithm first enumerates all possible allowable intervals $(i, j]$ as base cases (i.e., $(1 - \varepsilon)\sigma \leq j - i \leq (1 + \varepsilon)\sigma$), and computes $\text{LF}((i, j], i, i, i)$ for each such $(i, j]$. Then, for general $(i, j]$, the algorithm computes $\text{LF}((i, j], i_1, i_2, i_3)$ for all possible values of i_1, i_2 , and i_3 given i and j , in increasing order of $(i + j)$; this ensures all the intermediate subqueries are already computed before $\text{LF}((i, j], i_1, i_2, i_3)$ is evaluated. After it computes $\text{LF}((i, j], i_1, i_2, i_3)$ for all possible values of i, j, i_1, i_2, i_3 , it examines whether $\text{LF}((0, n], i_1, i_2, i_3) = \text{True}$ for some i_1, i_2, i_3 ; any such true entry implies a fair partition of $[n]$, i.e., the original input. Note that $(0, n] = [1, n]$ is the complete set of points.

Running time and Enhancements. For interval $(i, j]$ to belong to the base cases, it must hold that $j \in [i + (1 - \varepsilon)\sigma, i + (1 + \varepsilon)\sigma]$, for which $i \in [0, n - (1 - \varepsilon)\sigma]$. Hence, there are $O(n\sigma)$ such pairs. For each such interval $(i, j]$, whether or not it contains a deviating group as a standalone interval can be checked naively in $O(\sigma^3)$ -time. Hence all base cases can be computed within $O(n\sigma^4)$ -time.

For every general interval $(i, j]$, the algorithm needs to enumerate all possible values of i_1, i_2 , and i_3 . Note that the ranges for all possible values of i_1, i_2 , and i_3 are exactly $2\varepsilon\sigma$, $4\varepsilon\sigma$, and $6\varepsilon\sigma$; hence the number of such 3-tuples are bounded by $O(\varepsilon^3\sigma^3)$. For each subquery

$\text{LF}((i, j], i_1, i_2, i_3)$, the algorithm needs to enumerate all $O(\varepsilon\sigma)$ possible values of i_4 ; for each i_4 , the lookups of previously computed subqueries incur $O(1)$ computation, whereas the check for fairness for $(i_4, j]$ can be done naively in $O(\sigma^2)$ -time. Hence the algorithm computes the values of all subqueries in $O(\varepsilon^4\sigma^4)$ -time for $\varepsilon \in [0, 1/3)$ and $O(\varepsilon^4\sigma^6)$ -time for $\varepsilon \geq 1/3$.

For the last step, the algorithm linear scans $\text{LF}((0, n], i_1, i_2, i_3)$ for all $O(\varepsilon^3\sigma^3)$ possible values of (i_1, i_2, i_3) , which is $O(\varepsilon^3\sigma^3)$ -time. Therefore, the total time complexity of the algorithm is given by

$$O(n\sigma^4) + O(\varepsilon^4\sigma^6) + O(\varepsilon^3\sigma^3) = O(n\sigma^5),$$

as $\sigma = O(n)$ and $\varepsilon = O(1)$. For $\varepsilon \in [0, 1/3)$, it is $O(n\sigma^4)$ with the second term being $O(\varepsilon^4\sigma^4)$.

Running time enhancement with precomputation. In the algorithm above, the naive check for deviating group within a standalone interval for base cases incurs redundant computations. Instead, with standard dynamic programming techniques, one can precompute in $O(n\sigma)$ -time for each allowable region $(i, j]$: (i) the number of blue points and the number of red points in $(i, j]$, i.e., $|(i, j] \cap B|$ and $|(i, j] \cap R|$; and thus (ii) the number of unhappy points in $(i, j]$ if $(i, j]$ is contained in a blue-majority (resp. red-majority) interval. These values can be stored using $O(n\sigma)$ -memory. When checking if a deviating group $(i', j']$ exists within $(i, j]$, note that both $(i', j']$ and $(i, j]$ need to be standalone allowable intervals; this gives $i' \in (i, i + 2\varepsilon\sigma]$ and $j' \in (j - 2\varepsilon\sigma, j]$. Therefore, the check can be done in $O(\varepsilon^2\sigma^2)$ constant-time lookups.

This idea is also useful for speeding up the step of checking whether the partition $\{(i_4, i_3], (i_3, i_2], (i_2, i_1], (i_1, j]\}$ is a fair partition of $(i_4, j]$. Again using standard dynamic programming techniques, similar to the above, one can precompute in $O(n\varepsilon^4\sigma^4)$ -time for each (i_1, i_2, i_3, i_4, j) whether the above partition is fair (observe that there are $O(n)$ possibilities for j , then at most $2\varepsilon\sigma$ possibilities for i_1 , then at most $4\varepsilon\sigma$ possibilities for i_2 , and so on), such that the check can be replaced by $O(1)$ -time lookups when evaluating each

$\text{LF}((i, j], i_1, i_2, i_3)$.

For $\varepsilon \in [0, 1/3)$, only the first enhancement is needed, for which the total time complexity of the algorithm is improved to

$$O(n\sigma) + O(n\varepsilon^2\sigma^3) + O(\varepsilon^4\sigma^4) + O(\varepsilon^3\sigma^3) = O(n\sigma^3);$$

for $\varepsilon \in [1/3, 1/2]$, with both enhancements, the total time complexity of the algorithm is improved to

$$O(n\sigma) + O(n\varepsilon^4\sigma^4) + O(n\varepsilon^2\sigma^3) + O(\varepsilon^4\sigma^4) + O(\varepsilon^3\sigma^3) = O(n\sigma^4),$$

as $\sigma = O(n)$ and $\varepsilon = O(1)$.

Theorem 30. *Given an instance (X, σ) with $|X| = n$ and $\sigma \in [n]$, and parameters $\varepsilon \in [0, 1/2]$ and $\beta \in [1/2, 1]$, a (ε, β) -locally fair partition of $[n]$ can be computed, or report that none exists, in time $O(n\sigma^3)$ for $\varepsilon \in [0, 1/3)$ and $O(n\sigma^4)$ for $\varepsilon \in [1/3, 1/2]$.*

3.6 Conclusion

We note that many of our results do extend to the fixed k version. The existence results in Section 3.3 extend to the fixed- k case naturally. The dynamic program of Section 3.5 can be extended to handle a fixed k in a straightforward manner.

The main open question is extending the model to two dimensions, which poses several challenges: how to define feasible regions, how these regions tile the plane, how a deviating group/region is defined. Part of the difficulty stems from the fact that points are no longer linearly ordered. Although Voronoi region based methods have been proposed to construct compact regions, incorporating fairness seems very hard, and none of the existing algorithms extend to this case even to construct an approximate solution for clustered inputs. Currently, no polynomial-time algorithm is known even if each region of the partition is restricted to be an axis-aligned rectangle. In contrast, the additional freedom when defining two dimensional regions suggests a fair partition exists for a larger value of parameters assuming the input points are not concentrated along a 1D curve. Despite the

challenges, the lower bounds presented in the chapter directly extend to two dimensions. Additionally, the algorithm described in Section 3.5 may extend to 2D when the partitions considered have sufficient structure, such as when we restrict to a hierarchical partition of simple shapes, and the deviating region spans $O(1)$ regions of the partition. In the next chapter, we consider local fairness for redistricting US states, using the standard approach of modeling each state as a planar graph.

Chapter 4

Redistricting via Local Fairness

4.1 Introduction

In this chapter, we focus on the redistricting problem as a practical application. To recall, redistricting in the United States is the process of partitioning a state into districts, each of which elects one representative to the Congress, for the most part, via simple majority voting. As of April 2022, one year after the US Census Bureau released the results of the 2020 decennial census, 41 out of the 50 states have finished redrawing the congressional redistricting plans for the next decade [Lev22]. This process has triggered numerous debates and litigation along the way. Much of this debate centers on whether the plans are *gerrymandered* so that one of the two parties gets more representatives. Given its high-stakes impact and mathematical richness, there has been persistent interest in tackling redistricting as an algorithmic question since the early 1960s [BS20].

There is ongoing debate around what a “desirable” redistricting plan should be. It is commonly agreed that “desirable” plans should, at minimum, produce population balanced, contiguous, and compact districts [Rey64]. Beyond this basic agreement, there is still debate on richer notions of desirability, particularly notions related to the “fairness” of a plan. This has motivated a long line of recent work [DDS21b, DDS20, HRM17] as well as software tools [Pro22, Lev22] on *auditing* a given redistricting plan against various fairness concepts. Some of these concepts have since been adopted in Wisconsin’s and Michigan’s redistricting efforts [Che21]. It should be noted that under most notions of desirability proposed in literature, the problem of redistricting is computationally hard [KMV19], leading to the study of heuristic approaches, which we outline later.

Global versus Local Fairness. Zooming into fairness criteria, most extant notions of fairness focus on the *global outcomes* of the redistricting plans, e.g., whether the *seat shares*

proportionally represent the demographics [War18], or how competitive the districts drawn are [DDS20]. However, it is argued in [AKMT22] that global metrics do not always distinguish between *natural* gerrymandering – when the distribution of voters unavoidably prohibits certain globally fair outcomes – and *artificial* gerrymandering – when the plans are manipulated to favor a demographic group. This issue is typically addressed via *statistical tests* [DDS21b]: a probabilistic method is used to generate an ensemble of population balanced, contiguous, and compact plans, and the global fairness score in question is computed for each of these plans, yielding a histogram of scores. The plan in question is deemed “fair” if its global fairness score is not an outlier in this histogram.

Global fairness, such as proportional seat shares, despite being desirable and statistically testable, may not represent the *local* concerns of voters. For instance, imagine the blue party cares about rising sea levels and climate change, while the red party does not. In North Carolina, if at least one seat on the eastern coast has blue majority, that representative may advocate to mitigate the impacts of climate change to the coastal residents on the state or federal level. On the other hand, a better seat share may lead to a plan in which all districts near the coast have red majority, while the districts in the western mountains have blue majority. However, the latter set of representatives may not advocate for issues impacting the coastal residents, since it is not of local concern to the geographic area. This motivates the need for local fairness as a separate fairness measure, capturing at some level the saying “all politics is local”.

Borrowing the notion of core from cooperative game theory, the work of [AKMT22] defines local fairness notion as follows: given a redistricting plan, a voter is unsatisfied if the majority demographic in her district does not match her own demographic. A redistricting plan is locally fair if no group of unsatisfied voters could *deviate* and draw a different district such that this group of unsatisfied voters has a majority in the new district.

As in the scenario above, such a local notion of fairness has the advantage of capturing *justified complaints* of groups of voters, as has happened in earlier court judgements [Coo17]. It also provides a way of auditing enacted plans without resorting to statistical tests, making

it more human interpretable and *explainable*.

Research Questions. The notion of local fairness is appealing; however, the analysis and results in [AKMT22] are theoretical and apply only to a simplified one-dimensional model. In this chapter, we develop algorithms to audit plans for local fairness, and systematically study this concept on real-world electoral data. In particular, we study the following questions:

- Given a redistricting plan, can we efficiently test (or audit) whether the plan is locally fair?
- Are locally fair plans achievable in real redistricting tasks? If not, can we quantify how far a given plan is from being locally fair?
- Is local fairness empirically compatible with other existing global fairness concepts?

4.1.1 Related Work

Redistricting as Optimization. We first focus on the task of drawing plans, or computational redistricting. The idea of using computational tools in redistricting dates back to the 1960s [Vic61, HWS⁺65]. Since then, an extensive line of work (see [BS20] for a comprehensive survey) cast the redistricting task as an optimization problem, in which the input contains only spatial location of individuals, but not their political affiliations. The objective and constraints capture the population balance, contiguity, and compactness criteria of the districts. This problem is computationally intractable in the worst case [CKM⁺21], and multiple algorithmic approaches have been proposed, including Voronoi diagrams [LF19, GKM⁺21], local search [KJS15], simulated-annealing and hill climbing [AM11], and spatial evolutionary algorithms [LCW16b]. On the flip side, it is argued in [CR13, WK20] that such “neutral” districting plans – as outputs of algorithms without political inputs – may contain unintentional biases, as well as unexpected outcomes such as “natural gerrymandering” [BLSS18, GGRS22], i.e., the geographic distributions of voters naturally lead to disproportionate seat shares. Therefore, fairness objectives such as partisan representativeness

are typically incorporated into the redistricting problem as objectives; however, these additional requirements further add to the computational difficulty of the problem [KMV19].

Ensemble Approaches to Redistricting. Instead of optimizing and finding a single best redistricting plan, another line of work focuses on generating a large *ensemble* of districting plans, with the hope of some of these plans being fair. These methods include Flood Fill [CDO00, MM18], Column Generation [GS21], and the widely adopted Markov Chain Monte Carlo (MCMC) approach [TCL16, FHIT20, LCC⁺22]. The latter approach *samples* from the space of feasible plans with a bias towards “desirable” or fairness properties. For instance, it is shown in [PTF22] that the widely used ReCom MCMC method [DDS21b] provably biases towards compact plans. We provide a more in-depth description of ReCom in Section 4.4. The work of [EGB22] proposes a method for choosing one representative plan from such an ensemble based on defining distances between plans.

Auditing and Combating Gerrymandering. A somewhat different question from constructing a desirable plan is the question of *auditing* a given plan for desirability and fairness. As mentioned before, ensemble based approaches provide a natural, statistical way of auditing [HRM17, HKL⁺20]: The properties of the enacted plan is compared against the histogram of the corresponding property on the ensemble; if the plan is a statistical outlier, then it is considered more “gerrymandered” and hence less desirable. The recent work of [LCC⁺22] instead uses plans in the ensemble as comparators to identify manipulation in redistricting plans. On the non-statistical side, numerous approaches to auditing have also been proposed via appropriate desirability scores. These are either scores based on compactness of the plan (such as the Reock [Reo61] and Polsby-Popper [PP91] scores), or scores based on partisan outcomes generated by the plan (such as the efficiency gap [SM15], mean-median gap [Wan16], partisan symmetry [War18], and the GEO metric [CRSV22]), or scores based on competitiveness of the plan [DDS20]. Many of these measures are used in publicly available tools [Pro22, CRSV22]. Finally, there is a recent line of work that attempts to eliminate gerrymandering by completely revamping the winner-takes-all, single-member district mechanism into a multiwinner election [GGRS22].

4.1.2 Our Contribution

In this chapter, we take the standard view of redistricting as partitioning a planar graph on precincts into population-balanced, contiguous, and (in a heuristic sense) compact regions. We naturally extend the local fairness concept proposed in [AKMT22] to this task.

We first focus on the question of *auditing* a given plan for local fairness, that is, the non-existence of a population-balanced contiguous region in which a majority of voters are of the same party and is minority in the given plan. We show that this problem is computationally intractable in the worst case. Our first contribution is two heuristics for the auditing problem. Our first approach, that is scalable and practical, extends existing ensemble-based methods in a novel way: we assume the districts in the ensemble are the only districts to which voters can deviate, and given a plan to be audited, we test each of these districts as a potential deviation on that plan. Our second approach drills deeper into plans where the ensemble based method finds no deviating group; indeed, if the method found a deviating group, the plan was already deemed not locally fair. On the former set, we generate several random spanning trees, and devise a polynomial time dynamic programming algorithm that audits each tree for local fairness. If any of these audits finds a deviating group, the original plan was not locally fair. The dynamic program is not as efficient as the ensemble-based method; however, we provide empirical evidence that the ensemble method suffices to deem a plan locally fair, and the dynamic program typically does not find additional compact deviating groups. Finally, for redistricting plans that are not locally fair, we propose a measure that quantifies the unfairness of the plans by the portion of population with a justified complaint.

As our second contribution, we empirically study the notion of local fairness on real data on recent elections in the US. We generate plans using the (by now) standard ReCom [DDS21b] ensemble method, and audit each plan for local fairness using the ensemble method, thereby producing an ordering of the plans via our unfairness measure. We empirically show that applying the criterion of local fairness prunes the space of candidate plans considerably, while still returning a set of potential candidates. Most global and

statistical notions of fairness fail to do such pruning, since they are endogenously defined relative to the order statistics on the ensemble. We further show that not only is local fairness *achievable* on real redistricting tasks, but it is also compatible with extant global fairness properties. Indeed, when we compare locally fair plans and those with many deviating groups, the former tend to be just as compact, have comparable seat share outcomes, and sacrifice only a small amount of competitiveness. Thus local fairness can be used as an additional fairness criterion in conjunction with a global fairness criterion. We also investigate robustness of the local fairness concept, and show that fair redistricting plans remain consistent across different elections used. We finally show visualizations of fair and unfair plans; in particular showing that the visualization of deviating groups makes the local fairness notion explainable.

Taken together, our results demonstrate local fairness as an effective *pruning criterion* for candidate redistricting plans while sacrificing little in other desired properties. We also note that in practice, there could be other considerations when choosing the “best” plan even among many locally-fair plans; we leave the question of choosing these considerations to policy makers.

4.2 Model and Preliminaries

In keeping with recent literature [DT18, DDS21b, CKM⁺21, GS21], the input to the redistricting problem is a planar connected graph $G = (V, E)$ where each vertex $v \in V$ represents an indivisible geographic unit (a precinct or a census block),¹ and an edge is placed between two vertices if they are geographically adjacent. Going forward, we refer to each $v \in V$ as a *precinct* and G as the *precinct graph*.

Redistricting Plans. For each precinct $v \in V$, let $\rho(v) > 0$ denote its population and let $\tau(v) \in [0, \rho(v)]$ denote the number of voters in v .² We let $\gamma(v) \in [0, 1]$ and $\beta(v) = 1 - \gamma(v)$

¹Typically, precincts are not split by redistricting plans [DDS21b].

²We assume that we know the exact number of people who cast a vote in each precinct, along with which candidate they voted for, such as is available for historical elections

denote the fraction of $\tau(v)$ who vote *red* and *blue*, respectively. Note that it is assumed each individual voter is exactly one of the two colors. For an arbitrary subset of precincts $W \subseteq V$, set $\rho(W) := \sum_{v \in W} \rho(v)$, $\tau(W) := \sum_{v \in W} \tau(v)$, $\gamma(W) := \frac{\sum_{v \in W} (\gamma(v) \cdot \tau(v))}{\tau(W)}$, and $\beta(W) := 1 - \gamma(W)$.

Definition 31 *k*-redistricting plan. A *k*-redistricting plan of $G = (V, E)$ is a partition of V into *k* pairwise-disjoint subsets $D_1, D_2, \dots, D_k \subseteq V$, called districts. Each district assumes the color of the majority of its voters. For a redistricting plan Π , let B_Π (resp. R_Π) denote the set of precincts in blue (resp. red) districts in Π .

In the following, we fix an error parameter $\varepsilon > 0$, and the desired number of partitions *k*. Note that the average population per district is $\frac{\rho(V)}{k}$. We say a district $D \subseteq V$ is ε -feasible if: (1) D induces a connected subgraph, and (2) the population of D is at most ε away from average, i.e., $(1 - \varepsilon) \cdot \frac{\rho(V)}{k} \leq \rho(D) \leq (1 + \varepsilon) \cdot \frac{\rho(V)}{k}$. A redistricting plan Π is ε -feasible if each district $D_i \in \Pi$ is ε -feasible.

We note that this definition of an ε -feasible plan is consistent with the general practice in the U.S, where the sizes of districts should be balanced in terms of their population, based on census information, not in terms of the number of eligible voters. Since ε and *k* will be fixed throughout, we drop the prefixes and refer to *k*-redistricting plans and ε -feasible districts as *redistricting plans* and *feasible* districts, respectively.

Local Fairness. We extend the notion of *local fairness* proposed by [AKMT22] to the graph-based redistricting problem. We say that a feasible district $W \subseteq V$ is *red-majority* (*red* in short) if $\gamma(W) \geq \beta(W)$, and *blue-majority* (*blue*) otherwise. We call this majority color as the *color* of W . Given a redistricting plan Π , any voter whose color agrees with the color of its assigned district in Π is deemed *happy* with respect to Π , and the remaining voters are *unhappy*.

Definition 32 *c*-locally fair. Given a feasible redistricting plan Π of G and a constant $c \in [1/2, 1]$, a feasible district $W \subseteq V$ is a *red c-deviating group with respect to Π* if W is red and at least a *c*-fraction of its voters are *unhappy red voters* in Π , or formally,

$\sum_{v \in W \cap B_{\Pi}} \gamma(v) \cdot \tau(v) > c \cdot \tau(W)$. A blue c -deviating group is defined analogously. We call a feasible redistricting plan Π of G c -locally fair if there are no red or blue c -deviating groups with respect to Π .

When $c = 1/2$, only a simple majority of voters in a deviating group must be unhappy. In this special case, we omit the prefix c by referring to red deviating groups, blue deviating groups, and locally fair redistricting plans. Throughout, we refer to locally fair redistricting plans as fair plans.

We are thus interested in the following two problems.

LF Auditing problem. Given a feasible redistricting plan Π and a parameter $c \in [1/2, 1]$, decide whether Π is c -locally fair.

LFP Generation problem. Given a precinct graph G and parameters ε, k and c , compute a feasible redistricting plan Π of G such that Π is c -locally fair, or report that none exists.

For a redistricting plan Π that is not locally fair, we quantify its degree of unfairness as follows. Consider all deviating groups of Π , and define the unfairness score of Π as the fraction of all voters that are unhappy in some deviating group. Formally, let

$$W_{\text{B}}^*(\Pi) := R_{\Pi} \cap \left(\bigcup \{W \subseteq V \mid W \text{ is a blue deviating group of } \Pi\} \right)$$

denote the set of red precincts that lie in some blue deviating group of Π . Similarly, define $W_{\text{R}}^*(\Pi)$. Then the unfairness score of Π is defined as:

$$\text{unf}(\Pi) := \frac{\beta(W_{\text{B}}^*(\Pi)) \cdot \tau(W_{\text{B}}^*(\Pi)) + \gamma(W_{\text{R}}^*(\Pi)) \cdot \tau(W_{\text{R}}^*(\Pi))}{\tau(V)}.$$

This score captures the fraction of voters that are both (i) unhappy in Π and (ii) in the majority color of some deviating group of Π . Note that $\text{unf}(\Pi) \in [0, 1]$, and equals zero if Π is locally fair.

Compactness. In addition to requiring that districts be contiguous and population balanced, many redistricting models also require that the districts be *compact*. However, in

contrast to the former two criteria, there is no universally agreed measure of compactness [BS20, GS21]. For example, the Princeton Redistricting Report Cards³ uses Reock [Reo61] and Polsby-Popper [PP91] scores, both of which are derived from the area and perimeter of the geographic districts drawn. It also uses the number of counties split into multiple districts. In the discrete model of precinct graphs, one common measure of compactness is the number of cut edges formed by the plan, i.e., the number of edges whose endpoints lie in different districts of Π [DDS21b, CKM⁺21]. Though we do not enforce compactness in the generation and audit problems, the algorithms we use are biased towards compact plans, as we empirically demonstrate.

Organization. In Section 4.3, we show both the LF AUDITING and LFP GENERATION problems are NP-hard. In Section 4.4, we present two algorithms for the LF AUDITING and LFP GENERATION problems. We then describe the experimental setup and empirical results in Section 4.5. Additional methodological and experimental results are presented in the Supplementary Material.

4.3 Hardness

In this section, we show that both LF AUDITING and LFP GENERATION are NP-complete.

Theorem 33. LF AUDITING is NP-complete.

Proof. We reduce the NP-complete Connected k -Subgraph Problem on Planar Graphs with Binary Weights (CkS -PB) [HP94] to the LF AUDITING problem. Given a connected planar graph $G = (V, E)$ where $V = \{v_1, v_2, \dots, v_{|V|}\}$, a vertex weight $\omega(v_i) \in \{0, 1\}$ for each $v_i \in V$, a size $M \geq 2$, and a targeted total weight $\Omega \in \mathbb{Z}^+$,⁴ the decision version of CkS -PB asks whether there is a subset $W \subseteq V$ of M vertices such that its induced subgraph $H \subseteq G$ is connected and $\sum_{v_i \in W} \omega(v_i) \geq \Omega$. Here it is assumed that $M \leq |V|$ and $\omega(v_i) < \Omega$ for each $v_i \in V$; otherwise the problem is trivial.

³<https://gerrymander.princeton.edu/>

⁴We use M and Ω here to avoid confusing with the notations k and W in our problem.

Given an arbitrary instance of CkS-PB, we construct an instance for LF AUDITING as follows. For each vertex $v_i \in V$, we construct two precinct nodes v_i and u_i , and an edge between v_i and u_i . For each edge $(v_i, v_j) \in E$, we construct an edge between v_i and v_j . Hence, the resulting graph $G = (V, E)$ has $2|V|$ precinct nodes and $(|V| + |E|)$ edges, and is still planar.

For all $i = 1, 2, \dots, |V|$, we let $\rho(v_i) = \tau(v_i) = 2M\Omega$ and $\gamma(v_i) = \left(\frac{M-1}{2M} + \frac{\omega(v_i)}{2\Omega}\right)$; note that each $\gamma(v_i)$ is in $(0, 1)$ and each $\gamma(v_i) \cdot \tau(v_i)$ is an integer. For all i , we let $\rho(u_i) = \tau(u_i) = 2M(M-1)\Omega$, and $\gamma(u_i) = 0$. We call the precinct nodes v_i *regular* and the precinct nodes u_i *auxiliary*. Let $k = |V|$, $c = 1/2$, and pick ε such that $0 < \varepsilon < \frac{1}{M}$. Finally, let $\Pi = \{D_1, D_2, \dots, D_k\}$, where $D_i = \{v_i, u_i\}$ for all $i = 1, 2, \dots, k$. Observe that we have $\beta(D_i) > \frac{1}{2}$ for all $i = 1, 2, \dots, k$, i.e., we have $B_\Pi = V$ and $R_\Pi = \emptyset$. Note that since $0 < \varepsilon < \frac{1}{M}$, we have $(2M^2 - 2M)\Omega < (1 - \varepsilon)2M^2\Omega$ and $(2M^2 + 2M)\Omega > (1 + \varepsilon)2M^2\Omega$, and thus every feasible district has total population exactly $2M^2\Omega$.

Suppose the CkS-PB instance is a **Yes** instance, i.e., there is a subset $W \subseteq V$ of M vertices such that its induced subgraph $H \subseteq G$ is connected, and $\sum_{v_i \in W} \omega(v_i) \geq \Omega$. Let $W = \{v_i \mid v_i \in W\}$ be the set of regular precinct nodes corresponding to the vertices in W . Then we have $|W| = M$ and thus $\rho(W) = 2M^2\Omega$. Furthermore, we have

$$\begin{aligned} \sum_{v \in W \cap B_\Pi} \gamma(v)\tau(v) &= \sum_{v \in W} \gamma(v)\rho(v) = \sum_{i: v_i \in W} \left(\left(\frac{M-1}{2M} + \frac{\omega(v_i)}{2\Omega} \right) \cdot 2M\Omega \right) \\ &\geq \left(\frac{M-1}{2} + \frac{\Omega}{2\Omega} \right) \cdot 2M\Omega = M^2\Omega = c \cdot \tau(W). \end{aligned}$$

Since W induces a connected subgraph of G , it is a red c -deviating group of Π .

For the other direction, suppose the LF AUDITING instance is a **Yes** instance, i.e., there is a red c -deviating group W of Π . Suppose W contains at least one auxiliary precinct node u_i . Recall that $\rho(W) = 2M^2\Omega$. Hence, we have

$$\sum_{v \in W \cap B_\Pi} \gamma(v)\tau(v) = \sum_{v \in W} \gamma(v)\rho(v) < \sum_{v \in W \setminus \{u_i\}} \rho(v) = 2M\Omega \leq M^2\Omega = c \cdot \tau(W),$$

a contradiction. Hence W can only contain regular precinct nodes. Then we have $|W| = M$,

and

$$\begin{aligned}
\sum_{v_i \in W} \omega(v_i) &= \sum_{v \in W \cap B_\Pi} \left(2\Omega\gamma(v) - \frac{(M-1)\Omega}{M} \right) \\
&= \frac{1}{M} \cdot \sum_{v \in W} (\gamma(v)\rho(v) - (M-1)\Omega) \\
&\geq \frac{\rho(W)}{M} - (M-1)\Omega = M\Omega - (M-1)\Omega = \Omega.
\end{aligned}$$

Since W induces a connected subgraph of G , the corresponding CkS -PB instance is a **Yes** instance.

Combining the above, there is a polynomial-time reduction from CkS -PB to LF AUDITING. Since LF AUDITING is trivially in NP, we conclude that it is NP-complete. \square

We then observe that the same reduction also gives the hardness of LFP GENERATION.

Theorem 34. LFP GENERATION is NP-complete.

Proof. Observe that in each LF AUDITING instance constructed in the proof for Theorem 33, the associated plan Π is the only feasible redistricting plan. To see this, notice that every auxiliary node u_i has a single neighbor v_i and does not possess enough population to form a feasible district itself, and thus every u_i must be paired with its corresponding v_i to make a district. Therefore, the LF AUDITING and LFP GENERATION are identical on this family of instances, and the hardness of LFP GENERATION follows from the same reduction. \square

Remarks. We note that the proofs above still hold even if we add more edges to the constructed graph. More specifically, for both the proof of Theorems 33 and 34, we can safely add edges as long as (i) planarity is preserved and (ii) at least one of the endpoints of each additional edge is an auxiliary precinct node. To see this, observe that adding additional edges does not impact the feasibility of Π , and since valid deviating groups contain only regular precinct nodes, the set of possible deviating groups for Π remains identical. For the proof of Theorem 34, the additional edges may allow multiple feasible

redistricting plans, but each of the feasible redistricting plans must still contain k districts of one regular and one auxiliary precinct node each, and by the same reduction, either all or none of them are locally fair (corresponding to **Yes** and **No** CkS -PB instances).

Note further that in both **LF AUDITING** and **LFP GENERATION**, we do not explicitly require the districts and deviating groups to be compact with respect to any specific criteria. Under certain restrictive compactness constraints, the problems may become tractable. For example, if the districts and deviating groups are restricted to be subsets of precincts fully contained in a circle centered around a precinct point, then the set of possible districts and deviating groups has polynomial size, and thus **LF AUDITING** can be solved by enumeration in polynomial time.

4.4 Heuristics for Efficiently Auditing Local Fairness

We have shown that both the **LF AUDITING** and **LFP GENERATION** problem are NP-complete, thereby necessitating heuristic or approximately optimal approaches.

Our main contribution in this section is efficient heuristics for **LF AUDITING**. The methods trade off computational efficiency for accuracy in determining local fairness. We describe the types of inaccuracy (one-sided versus two-sided error) that arise after describing the methods. We also provide empirical evidence that the more computationally efficient of these methods suffices to deem plans as locally fair. For **LFP GENERATION**, we simply generate an ensemble of feasible redistricting plans using existing MCMC approaches, and run the **LF AUDITING** algorithm to find the unfairness score $\text{unf}(\Pi)$ for each generated plan Π , thereby ranking the plans in the ensemble in terms of fairness.

4.4.1 Ensemble-based Auditing

The computationally more efficient approach makes use of ensembles in a novel way as follows:

1. **Generation.** Generate an ensemble of t redistricting plans $\mathcal{E} = \{\Pi_1, \Pi_2, \dots, \Pi_t\}$.
2. **Districts to test.** Let $\Delta = \bigcup_{j=1}^t \Pi_j$ be the collection of districts in the plans in \mathcal{E} .

3. **Audit.** Treat Δ as the candidate set of deviating groups. For each plan $\Pi \in \mathcal{E}$, compute if each $D \in \Delta$ is a c -deviating group (either red or blue). This step yields, for each $\Pi \in \mathcal{E}$, the set $\tilde{W}_B^*(\Pi) \subseteq V$ of precincts in blue deviating groups $D \in \Delta$ (which we use as an approximation of $W_B^*(\Pi)$), and similarly $\tilde{W}_R^*(\Pi)$. We then use $\tilde{W}_B^*(\Pi)$ and $\tilde{W}_R^*(\Pi)$ to compute the unfairness score $\text{unf}(\Pi)$ for each $\Pi \in \mathcal{E}$.

For the first step of an ensemble of plans \mathcal{E} , we use the **ReCom** algorithm [DDS21b]. This algorithm develops a Markov Chain whose state space is the full set of (ε) -feasible (k) -partitions of G . In each step, it randomly combines two (or more in its general form) districts of the current redistricting plan, generates a random spanning tree of the subgraph induced by the combined districts, and re-partitions the subgraph into two parts by cutting edges in the spanning tree so that the new resulting districts remain population balanced. The random spanning tree step *biases* the Markov chain towards plans with compact districts, where compactness is measured by number of cut edges [PTF22].

In the remainder of the chapter, we describe our auditing framework when **ReCom** is used as the ensemble generation technique. We note however that we can use any other method that produces compact plans, for instance, iterative-merging flood-fill algorithms [CR13].

4.4.2 Auditing via Dynamic Programming on Trees

The ensemble-based LF AUDITING method makes one-sided error – if it finds a deviating group, then the plan is not fair, while the absence of a deviating group from Δ only means the plan is *likely* fair. Indeed, in our experiments, there are often multiple redistricting plans that the ensemble-based audit deems likely fair, calling for a more systematic method for further analyzing the candidate plans that are potentially fair. This motivates the approach we now describe.

We show that the LF AUDITING problem is efficiently solvable if G is a tree. We then exploit this fact and use this as a heuristic to solve LF AUDITING on a general planar graph G , given partition Π , as follows:

1. **Random spanning trees.** First generate a collection of random spanning trees \mathcal{T} of G .
2. **Audit each tree.** For each tree $T \in \mathcal{T}$, decide if there exists a feasible deviating group with respect to Π such that this district is a connected subtree of T . Note that such a district will also be a connected subgraph in G .

We implement step 2 for a tree T using dynamic programming. We check for the existence of a blue deviating group for $c = 1/2$ in a tree T as follows; the case for red groups and larger c follows similarly. Note that the range for the size of a district is $[(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$, where $\sigma = \rho(V)/k$. For each precinct $v \in V$, let $D(v)$ denote the district that contains v in Π . Let the number of unhappy blue individuals in precinct v be denoted $\text{uhp}(v) = \beta(v) \cdot \tau(v)$ if $D(v) \in R_\Pi$, and zero otherwise. The task is to decide whether there is a connected subtree $W \subset T$ such that $\sum_{v \in W} \rho(v) \in [(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$ and $\sum_{v \in W} \text{uhp}(v) > \frac{1}{2} \cdot \sum_{v \in W} \tau(v)$.

For each $v \in V$, let T_v be the subtree of T rooted at v . For a vertex $v \in V$ and two parameters $i, p \leq (1 + \varepsilon)\sigma$, let $A[v, i, p]$ denote the maximum number of unhappy blue voters in a subtree $W \subseteq T_v$ such that $v \in W$, $\tau(W) \leq i$, and $\rho(W) = p$. To compute $A[v, i, p]$, we take the best subset of children of v such that unhappy blue voters are maximized subject to population and voter constraints.

For a leaf v of T , $A[v, i, p] = \text{uhp}(v)$ if $i \geq \tau(v)$ and $p = \rho(v)$, and $A[v, i, p] = 0$ otherwise.

Next, let v be an internal node of T . Let $U = \text{children}(v) = \{u_1, \dots, u_{\deg(v)}\}$ denote the set of v 's children in T . Let $i' = i - \tau(v)$ and $p' = p - \rho(v)$. We have

$$A[v, i, p] = \text{uhp}(v) + \max_{\{i_j, p_j\}} \left\{ \sum_{j=1}^{\deg(v)} A[u_j, i_j, p_j] \mid \left(\sum_{j=1}^{\deg(v)} i_j \leq i' \right) \wedge \left(\sum_{j=1}^{\deg(v)} p_j = p' \right) \right\}. \quad (4.1)$$

To compute the second term in Equation (4.1) efficiently, let $B_{v,i,p}[j, x, y]$ denote the maximum number of total unhappy blue voters in a union of subtrees rooted at $\{u_1, \dots, u_j\}$ with total voter count at most x and total population level y . The second term of the RHS

in Equation (4.1) is $B_{v,i,p}[\deg(v), i', p']$. We then have $B_{v,i,p}[1, x, y] = A[u_1, x, y]$ for all x, y , and for all $j \geq 2$ we have

$$B_{v,i,p}[j, x, y] = \max_{x' \in [0, x], y' \in [0, y]} \{B_{v,i,p}[j-1, x-x', y-y'] + A[u_j, x', y']\}. \quad (4.2)$$

The algorithm thus proceeds by computing all $A[v, i, p]$ in an outer loop in increasing order of i and p and in bottom-up order of v . Each computation of $A[v, i, p]$ proceeds with an inner loop that computes $B_{v,i,p}[j, x, y]$ in increasing order of j, x , and y . For each v , after computing all $A[v, i, p]$, we check if there is any $p \in [(1-\varepsilon)\sigma, (1+\varepsilon)\sigma]$ such that $A[v, i, p] > \frac{i}{2}$. If so, there exists a blue deviating group rooted at v with population p , total voter count at most i , and total number of unhappy blue voters $A[v, i, p]$.

Time complexity. The algorithm computes $O(|V| \cdot \sigma^2)$ values of $A[v, i, p]$, each computing $O(\deg(T) \cdot \sigma^2)$ values of $B_{v,i,p}[j, x, y]$. Computing each $B_{v,i,p}[j, x, y]$ requires $O(\sigma^2)$ -time to loop through all values of x' and y' . Hence the overall time complexity is $O(|V| \cdot \sigma^6 \cdot \deg(T))$. We conclude the following.

Theorem 35. LF AUDITING *problem on a tree* $T(V, E)$ can be solved in time $O(|V| \cdot \sigma^6 \cdot \deg(T))$. Here, $\deg(T)$ is the maximum degree of a node in T .

Ensemble-based Auditing is Approximately Sufficient The dynamic programming method is less desirable because of two significant limitations. The first is its running time, which is prohibitively high for practical use. The second limitation is the introduction of a different type of error from the ensemble-based method. Note that as with the ensemble-based method, the non-existence of a deviating group in the DP only provides high confidence (but not absolute certainty) about the plan being locally fair. At the same time, there is a more subtle type of error the DP can make in the other direction: even if it finds a deviating group on the tree, this may not be a *compact* deviating group in the original graph. Therefore, the algorithm can output “not locally fair” when the only deviating groups are non-compact. We therefore need a final step where we check the deviating group to see that it is both feasible and compact on the original graph.

4.4.3 Speeding up the DP and Sufficiency of Ensemble-based Auditing

Although our dynamic programming algorithm for solving LF AUDITING on trees run in polynomial time, the time complexity of the algorithm is prohibitively high to be efficient in practice. Our goal in this section is to empirically demonstrate that this approach is not needed in practice, that is, it does not find reasonable deviating groups on plans that the ensemble-based method deems locally fair, hence showing that the ensemble-based auditing method is sufficient and obviating the need for the computationally inefficient dynamic programming.

Towards this end, we first show that the dynamic program can be sped up significantly if (among other things) we *interpolate* the voter information to the entire population of a precinct, so that $\tau(v) = \rho(v)$ for each precinct v . After performing such interpolation on the data used in our experiments (Section 4.5), we first run the ensemble-based auditing method to find fair and unfair plans for $c = 0.5$. Next, for each of these plans, we run the dynamic program to find deviating groups, checking each one for compactness and the value c for which it is a c -deviating group. We show that the dynamic program is unable to find compact deviating groups with $c \geq 0.52$ on the ensemble-audited 0.5-locally fair plans (on the interpolated data). This demonstrates the sufficiency of ensemble-based auditing if we relax the strength c of the deviating group slightly.

We note that our main experiments in Section 4.5 use actual voter data, since it is unclear how such data should be interpolated in a principled way to the entire population. In the current section, we perform the interpolation in a simple way only to make the DP run efficiently, which in turn enables us to demonstrate the conceptual point that the ensemble-based method suffices. This provides strong evidence that even without interpolation, the ensemble-based method will suffice.

Improving Running Time of Dynamic Program We first describe our approach to speed up the running time of the dynamic program.

Special case of $\tau(v) = \rho(v)$. We first assume that $\tau(v) = \rho(v)$ holds for all $v \in V$, i.e., every individual is labeled red or blue in every precinct. In this case, we can reduce the state space by dropping the state variable p , that is, we let $A[v, i]$ denote the maximum number of unhappy blue voters in a subtree $W \subseteq T_v$ such that $v \in W$ and $\tau(W) = \rho(W) = i$. In this case, for leaf precincts v we have

$$A[v, i] = \begin{cases} \text{uhp}(v), & i = \tau(v) = \rho(v); \\ 0, & \text{otherwise,} \end{cases}$$

and for general precincts v (with children $\{u_1, \dots, u_{\deg(v)}\}$) we have

$$A[v, i] = \text{uhp}(v) + B_{v,i}[\deg(v), p - \rho(v)], \quad (4.3)$$

where $B_{v,i}[1, x] = A[u_1, x]$ for all x , and for all $j \geq 2$ we have

$$B_{v,i}[j, x] = \max_{x' \in [0, x]} \{B_{v,i}[j-1, x-x'] + A[u_j, x']\}. \quad (4.4)$$

Now, the algorithm computes $O(|V| \cdot \sigma)$ values of $A[v, i]$, each computing $O(\deg(T) \cdot \sigma)$ values of $B_{v,i}[j, x]$, each requiring $O(\sigma)$ -time to loop through all values of x' . The overall time complexity thus drops to $O(|V| \cdot \sigma^3 \cdot \deg(T))$.

Relaxing size of deviation. We next modify the state $A[v, i]$ to be the maximum number of unhappy blue voters in a subtree $W \subseteq T_v$ such that $v \in W$ and $\tau(W) = \rho(W) \leq i$, i.e., it is now allowed that the subtree W has an aggregate population of less than i . Note that this induces a potential one-sided error in checking for the existence of deviating groups. To see this, consider the case when the algorithm finds some (v, i) such that $i \in [(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$ and $A[v, i] > i/2$. Now this corresponds to a subtree W rooted at v with a population (or voter count) of *at most* i and a total number of unhappy blue voters of *at least* $i/2$. While this still ensures a majority of voters in W are unhappy voters of the same color, the actual population size may be less than i and thus outside of the acceptable range $[(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$. However, we observe that this error is one-sided: Suppose there is indeed a deviating group W of the correct population size $i \in [(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$ with a total

number of unhappy blue voters of *at least* $i/2$, our algorithm must either find W , or find a deviating group W' with population at most i and a larger number of unhappy blue voters. Therefore, if the algorithm does not find any deviating group under the relaxed definition, we can still conclude that there is no deviating group (with respect to the current spanning tree T).

Pruning the states. Under the modified semantics of $A[v, i]$, we observe that each $A[v, i]$ is non-decreasing in i and each $B_{v,i}[j, x]$ is non-decreasing in x . Now consider the computation of some fixed $B_{v,i}[j, x]$. We maintain an upper bound ub and a lower bound lb of $B_{v,i}[j, x]$. whenever $lb \geq ub$, we terminate the computation early and return $lb = ub$ as $B_{v,i}[j, x]$. Since the $B_{v,i}[j, x]$ are computed in increasing order of x , we initialize $lb = B_{v,i}[j, x - 1]$ and let $lb = 0$ if $x = 0$. We also initialize $ub = B_{v,i}[j - 1, x] + A[u_j, x]$.

When the max function in Eq. (4.4) is evaluated in increasing order of x' , we update:

- $lb \leftarrow \max\{lb, B_{v,i}[j - 1, x - x'] + A[u_j, x']\};$
- $ub \leftarrow \min\{ub, B_{v,i}[j - 1, x - x'] + A[u_j, x]\}.$

The second step is because that for any $x'' > x'$, we have

$$B_{v,i}[j - 1, x - x''] + A[u_j, x''] \leq B_{v,i}[j - 1, x - x'] + A[u_j, x].$$

Therefore, $B_{v,i}[j - 1, x - x'] + A[u_j, x]$ is the maximum possible value of $B_{v,i}[j, x]$ if the function is maximized at any $x'' > x'$. If this is matched by lb , then the final maximum value will be exactly lb . In this case, we terminate the computation without examining any $x'' > x'$ in Eq. (4.4).

The same idea is applied to the computation of $A[v, i]$: We maintain a lower bound lb' for $A[v, i]$ (initialized to $A[v, i - 1]$), and whenever

$$lb' \geq ub' = \text{uhp}(v) + \sum_{j=1}^{\deg(v)} A[u_j, i'],$$

we return $A[v, i] = lb'$.

Rounding the population. For a fixed threshold parameter P , we round each $\rho(v)$ down to the largest multiple of P that is smaller than or equal to $\rho(v)$. Formally, let $\rho'(v) = \left\lfloor \frac{\rho(v)}{P} \right\rfloor \cdot P$. For any subtree $W \subseteq T$, we have $\sum_{v \in W} \rho'(v) \leq \sum_{v \in W} \rho(v)$.

Let $A'[v, i]$ denote the output of the algorithm when the rounded population level $\rho'(v)$ is used instead of $\rho(v)$. Then we have $A'[v, i] \geq A[v, i]$. Therefore, running our algorithm with rounding introduces one-sided error: If there exists any deviating group W of Π population level i , then the algorithm with rounding can also output W since it has the same number of unhappy blue voters with a rounded-down population level. We must then relax the acceptable size range from $[(1 - \varepsilon)\sigma, (1 + \varepsilon)\sigma]$ to $[(1 - \varepsilon)\sigma - P \cdot |V|, (1 + \varepsilon)\sigma]$ to accommodate this error and so that W becomes a candidate deviating group.

Final running time. With all these strategies incorporated, the running time of the dynamic program is reduced to $O(|V| \cdot (\frac{\sigma}{P})^3 \cdot \deg(T))$ as there are now only $O(\frac{\sigma}{P})$ population levels. This is the version of the algorithm that we implement.

4.5 Experiments

In our experiments, we attempt to answer the following questions. First, is local fairness achievable on real redistricting tasks? Second, is it compatible with extant measures of global fairness? Finally, is the notion robust if the underlying data changes? Given the previous discussion, our experiments in this section focus exclusively on the ensemble based method. In Section 4.4.3 we discuss the experiments for auditing via dynamic programming.

4.5.1 Datasets and Methods

All data used in our experiments is obtained from the MGGG States open repository [MGG22]. We obtain shapefiles and precinct graphs for Massachusetts (MA), Maryland (MD), Michigan (MI), North Carolina (NC), Pennsylvania (PA), Texas (TX), and Wisconsin (WI).⁵We

⁵These are chosen to represent a spectrum from states whose elections are typically competitive (e.g., NC, WI) to states whose elections are typically lopsided (e.g., MA, TX).

Table 4.1: Properties of data and ensembles.

State	$ V $	$ E $	$\rho(V)$	$\tau(V)$	k	$ \Delta $
MA	2.1K	5.9K	6.55M	5.13M	9	8.5K
MD	1.8K	4.7K	5.77M	4.42M	8	7.8K
MI	4.8K	12.5K	9.88M	7.54M	14	14.0K
NC	2.7K	7.6K	9.53M	7.25M	13	12.9K
TX	8.9K	24.7K	25.14M	18.28M	36	35.9K
PA	9.2K	25.7K	12.70M	9.91M	18	18.0K
WI	7.1K	19.5K	5.69M	4.35M	8	7.87K

set $\rho(v)$ to the 2010 census total population in each precinct v . The default election we use is the 2016 presidential election, while the 2012 presidential election is also used in the robustness tests. For each precinct v , we collect the number of total votes for the Republican party r_v and the total vote amount for the Democrat party b_v in the 2016 presidential election from [MGG22]. We set $\gamma(v) = r_v/(r_v + b_v)$, $\beta(v) = 1 - r_v$, and $\tau(v) = r_v + b_v$, so $\tau(v)$ is the total number of red (Republican) and blue (Democrat) voters.

Using the ReCom algorithm, we generate an ensemble of redistricting plans for each state.⁶ Each ensemble consists of 1,000 redistricting plans, each being the outcome of an independent 10,000-step Markov chain (with default population balance parameter $\varepsilon = 0.02$) seeded with a recent congressional electoral plan of the state. We set k to be the number of congressional districts in the 2016 election in each state. We then obtain the collection Δ of candidate districts for each state by taking the union of the districts in each plan in the ensemble. The properties of input graphs of states and their corresponding ensembles are summarized in Table 4.1.

Table 4.2: Percent of plans without c -deviating groups for different c values.

State	MA	MD	MI	NC	PA	TX	WI
$c = .5$	100%	26.7%	0.3%	4.9%	12.0%	2.8%	76.9%
$c = .51$	100%	27.0%	1.6%	8.7%	28.6%	5.4%	83.6%
$c = .52$	100%	28.0%	11.9%	14.8%	41.4%	8.7%	87.2%
$c = .55$	100%	31.5%	35.2%	63.9%	95.2%	26.0%	94.0%

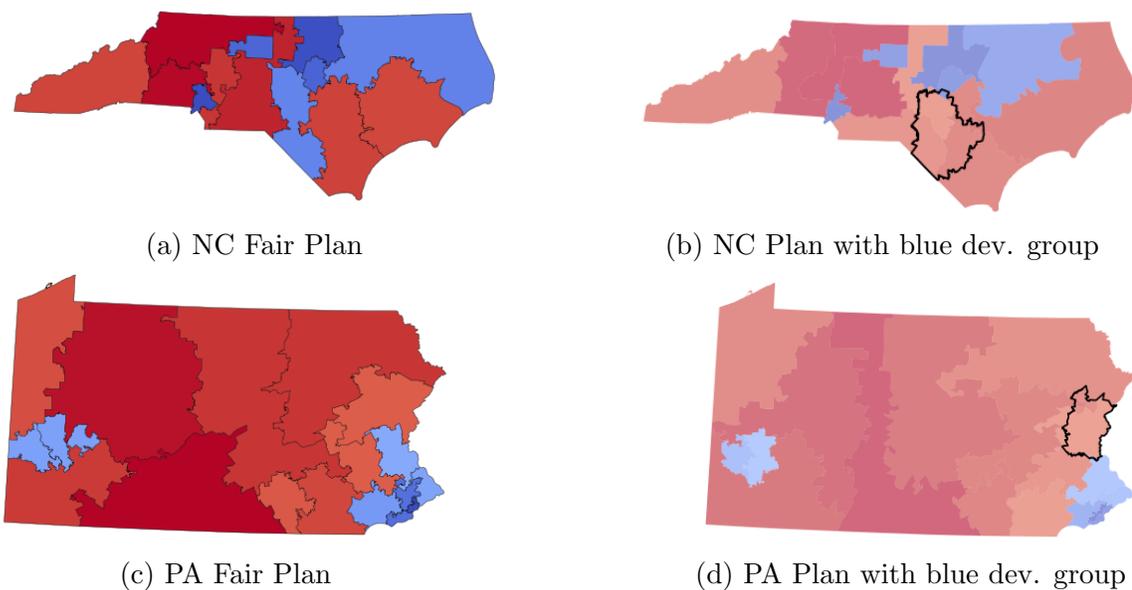


Figure 4.1: North Carolina and Pennsylvania plans without and with deviating groups. The blue deviating groups are drawn with a black outline. In these figures, the districts are coded by its color and the extent of partisanship: districts with a larger value of γ (resp. β) are colored in darker red (resp. blue).

4.5.2 Locally Fair Plans: Counts and Visualizations

We first use the ensemble-based audit method to audit each plan in the ensemble against the set Δ of districts from the entire ensemble. For each $c \in \{0.5, 0.51, 0.52, 0.55\}$, we count the number of plans in the ensemble without c -deviating groups in Δ . We present the results in Table 4.2. For all values of c , there exists plans in the ensemble that admit no c -deviating groups in Δ and thus are identified locally fair by the ensemble-based algorithm. Clearly, a larger value of c implies more plans are identified as locally fair. With $c = 1/2$, very few (but a non-zero number of) plans are identified as fair in four of the seven states. Every plan in the MA ensemble is identified as fair with all districts won by one party. Hence, we omit MA from all subsequent experiments.

In Figures 4.1a and 4.1c, we present examples of fair plans (with no 0.5-deviating groups) in NC and PA respectively, while in Figures 4.1b and 4.1d, we present “unfair” plans with many 0.5-deviating groups. We show visualizations for other states in Section 4.8.

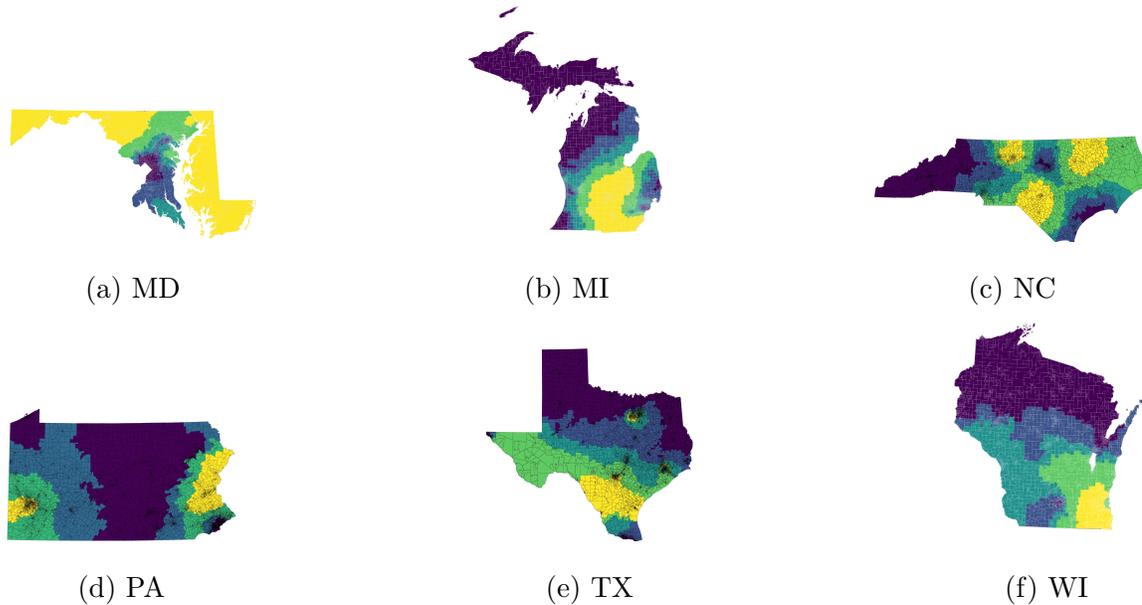


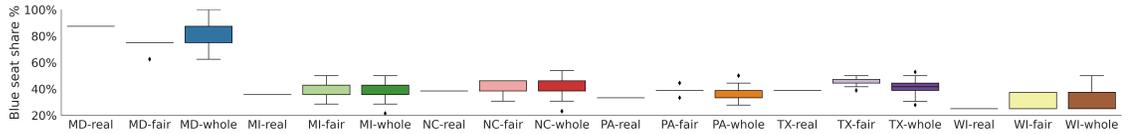
Figure 4.2: Precincts shaded in lighter color are contained in more deviating groups.

We note that a large fraction of plans are locally fair in some states (WI), while others

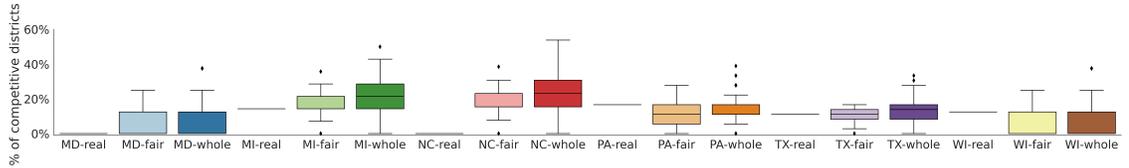
⁶Note that ReCom generates plans without taking into account electoral data.

have many deviating groups, even for a large setting of c (MI). To understand where deviating groups are located, in Figure 4.2, we plot a heat-map of the precincts, counting the number of 1/2-deviating groups (of either color) that contain that precinct, with yellow representing large counts and purple representing low counts. In every state except MD,⁷ we observe that precincts with highest counts are located either in an urban area or in proximity of one. This phenomenon is consistent with the perceived correlation between voter distribution and type of residence [BLSS18, WK20]. In states with multiple dense urban areas (e.g., NC), there is sufficient flexibility in the redistricting process to “crack” a highly-concentrated urban demographic into multiple districts. In this case, the urban area may form a deviating group resulting in high numbers of unfair plans.

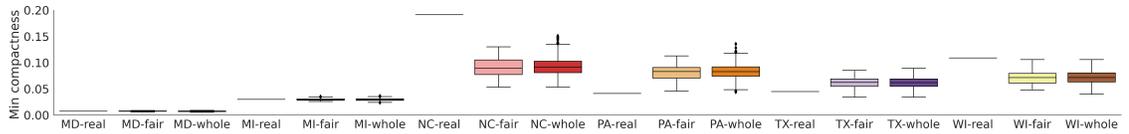
We note that our visualizations – in particular, the deviating groups in unfair plans, as well as the heat-map of likelihood of unhappiness of a precinct – make the local fairness concept *explainable*.



(a) Seat shares as percent of districts with blue majority.



(b) Percentage of districts being competitive (having less than 53.5% voters in majority).



(c) Minimum Polsby-Popper score of districts in a plan.

Figure 4.3: Distribution of fairness and compactness metrics among subsets of generated plans

⁷We discuss MD in more detail in Section 4.8.

4.5.3 Compatibility with Extant Fairness and Compactness Notions

The ensemble-based auditing approach can be viewed as a pruning method that identifies a subset of plans as locally fair. We now ask: how do locally fair plans perform on extant global fairness and compactness criteria compared to average plans in the ensemble? Towards this end, for $c = 1/2$, we rank the plans in the ensemble by the unfairness score $\text{unf}(\Pi)$. We compare properties of the top 5% plans in the ranking (which are most locally fair) against the entire ensemble, as well as a recent enacted congressional redistricting plan.⁸

Seat share outcomes. For each plan, we compute *Blue%*, the percentage of seat shares claimed by the Democratic candidate. The resulting distribution is shown in Figure 4.3a. The seat share distributions of the top 5% locally fair plans are comparable to the entire ensemble, sometimes achieving lower variance.

Number of competitive districts. The Princeton Gerrymandering Project [Pro22] defines a district as *competitive* if the majority color is at most 53.5% of the total votes. Following this definition, in Figure 4.3b, we compare the percent of competitive districts in a plan. The locally fair plans produce slightly fewer competitive districts: since larger majorities reduce the number of unhappy voters, finding a deviating group becomes harder. However, there exist fair plans that produce the median percentage of competitive districts in the ensemble in all but one state, and they are comparable with or better than the enacted plan in all states. We present results using a different competitiveness metric in Section 4.6.

Minimum compactness. Another measure of quality of districting plans is compactness – a non-compact district not only makes less geographic sense, but is also more likely to have been gerrymandered to favor one party over another. Two commonly used metrics to evaluate the compactness in redistricting plans are the average and minimum Polsby-

⁸If more than the 5% of the plans are locally fair, we take an arbitrary subset of the locally fair plans to serve as the top 5%.

Popper scores [PP91] of the districts in the plan [Pro22]: For a district $D \in \Pi$, the Polsby-Popper score is defined as $4\pi A(D)/P(D)^2$, where $A(D)$ and $P(D)$ are the area and the perimeter of the planar region D , respectively; a higher value implies a more compact district. In Figure 4.3c, we show that the minimum compactness of the locally fair plans remains comparable to that of the entire ensemble. We present results on the average Polsby-Popper score in Section 4.6.

Taken together, our results show that local fairness is compatible with fair seat share and compactness, while sacrificing only a small amount on number of competitive districts.

4.5.4 Actual Plans

We also compute the local fairness of plans actually enacted for previous elections. As it is relatively easy to find a locally fair plan in WI, its actual plan is indeed locally fair, while the actual plans for all other states are not locally fair.⁹Using the unf ranks, the enacted plan falls in the 55th percentile (fairer than 45% of the plans) in MD, 73th percentile in MI, 84th percentile in NC, 29th percentile in PA, and 22nd percentile in TX. While the MI and NC plans are (somewhat surprisingly) above average in local fairness, they have very few (or no) competitive districts. In general, enacted plans that achieve above average local fairness compared to the ensemble (MD, MI, and NC) have fewer competitive districts, and enacted plans with more competitive districts perform below average in local fairness. On the other hand, our results demonstrate that it is possible to find locally fair plans with a comparable (to the ensemble) amount of districts remaining competitive.

4.5.5 Sufficiency of Ensemble-based Auditing

We now use both our ensemble approach and the dynamic program to audit the ensemble for NC. We use the same experimental setup as in Section 4.5, except for one change. Since the DP assumes $\tau(v) = \rho(v)$, we need to interpolate the voter labels to the entire

⁹Note that all experiments use the 2016 presidential election data, while the plans in use are mostly drawn in 2011, except the NC one that is drawn in 2019.

precinct. We do this in the natural way. We keep the same $\gamma(v)$ and $\beta(v)$ values determined from an election, but let $\tau(v) = \rho(v)$. The number of red and blue voters in a precinct v become $\gamma(v) \cdot \rho(v)$ and $\beta(v) \cdot \rho(v)$, respectively. This is equivalent to assuming that in each precinct v , the rate of red/blue preferences of non-voters is identical to that of the voters. Accordingly, a c -deviating group must have a c fraction of the total population being unhappy individuals of the same color.

Using the interpolated voter labels on NC data, we first run the ensemble-based auditing method assuming $c = 0.5$. We find that 52 among the 1,000 plans (5.2%) in the ensemble do not have 0.5-deviating groups and are deemed fair by the ensemble approach. We again rank the plans in the ensemble by their unfairness score $\text{unf}(\Pi)$. We then construct two groups of plans: (1) 26 Plans deemed 0.5-fair by the ensemble approach; (2) 10 Plans in the bottom 5% (most unfair, in terms of unf score) in the ranking. We generate 5 random spanning trees of the NC precinct graph. For each plan and each spanning tree, we run the dynamic program where the population rounding parameter is set as $P = 750$. For each group of plans (fair and unfair), we obtain the set of all deviating groups found by the dynamic program on any spanning tree. We measure the Polsby-Popper compactness score of each deviating group on the original graph and the *strength* c for which the group is a c -deviating group in that plan (the largest c for which the group is indeed deviating). Note that a larger value of c implies the deviation is robust to small population changes, and is more significant in terms of unfairness.

In Figures 4.4a and 4.4b, we plot the heatmaps of the deviating groups found by the dynamic program for the fair and unfair sets of plans, respectively, where the x -axis and y -axis demonstrate their Polsby-Popper score and their *strength* respectively. As shown, for the plans deemed 0.5-fair by the ensemble approach, most deviating groups are either not compact (having low Polsby-Popper scores of < 0.1) or not strong (having strength values close to 0.5). As context, the minimum and average Polsby-Popper scores over all NC districts in the ReCom-generated NC ensemble are 0.053 and 0.177, respectively (corresponding to the two vertical lines in both plots); in other words, deviating groups

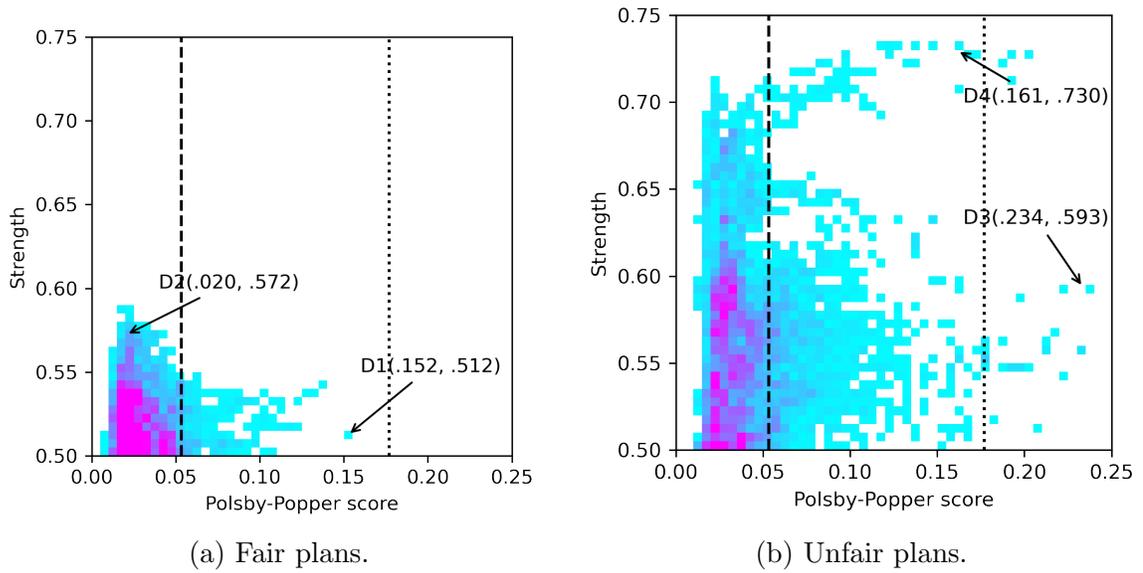


Figure 4.4: Heatmaps of deviating groups found by the dynamic program on fair and unfair NC plans.

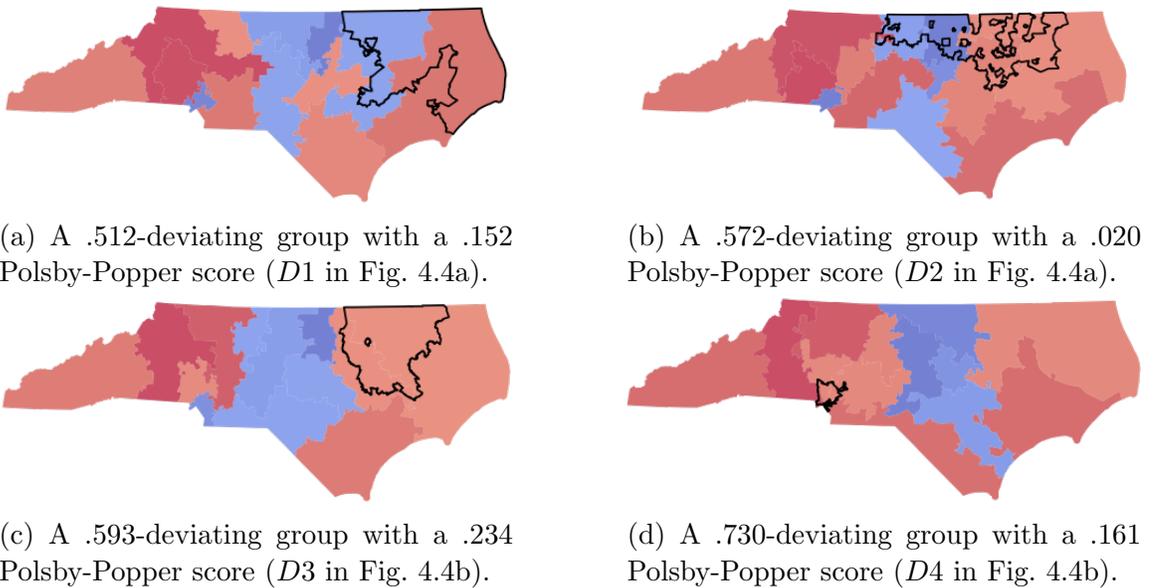


Figure 4.5: Deviating groups found by the dynamic program for two fair plans (left two maps) and two unfair plans (right two maps).

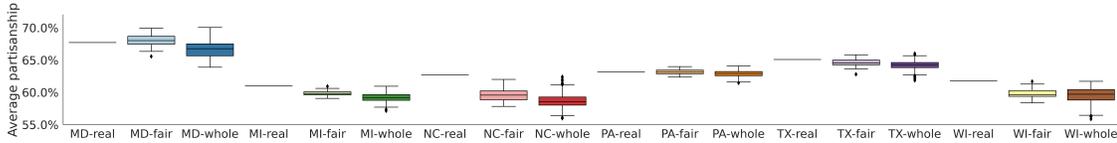
with Polsby-Popper scores of < 0.053 (to the left of the dashed vertical line) are less compact than *every one* of the 10k districts in the ensemble.

In fact, our dynamic program finds no deviating group with an above-average Polsby-Popper score for any 0.5-fair plan. The closest deviating group, shown as D1 in Figure 4.4a, has Polsby-Popper score 0.152 and a strength of 0.512; we visualize it in Figure 4.5a. In Figure 4.5b, we visualize another deviating group found by the dynamic program (shown as D2 in Figure 4.4a) with a Polsby-Popper score of 0.020 and a strength of 0.572. As manifested in the visualizations, deviating groups with very low Polsby-Popper scores like 0.02 are spurious with artificial shapes (such as holes) and should not be considered when it comes to determining whether a redistricting plan is fair. All the deviating groups for the 0.5-fair plans with a Polsby-Popper score at least 0.053 have lower strength (< 0.55). In summary, we can reasonably conclude that most of the fair plans found by the ensemble-based auditing approach do not admit strong, contiguous, and reasonably compact deviating groups even when audited by the dynamic program.

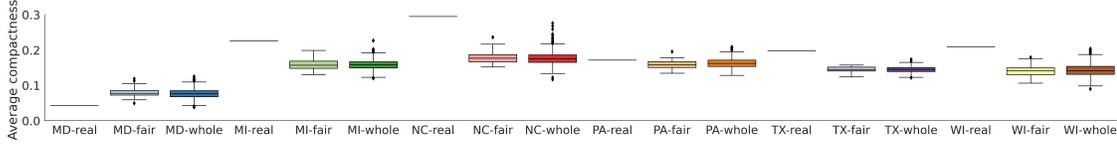
In contrast, for the plans ranked in the bottom 5% according to the ensemble-based approach, the dynamic program is able to find both strong and compact deviating groups quite easily. In Figures 4.5c and 4.5d we show (a) a .593-deviating group with a .234 Polsby-Popper score (D3 in Figure 4.4b), and (b) a .730-deviating group with a .161 Polsby-Popper score (D4 in Figure 4.4b) that the dynamic program find for one of the unfair plans. These results show that the strengths of deviating groups for fair plans (according to ensemble based auditing) are considerably lower than that for unfair plans. In other words, the results via DP validates that via the ensemble based approach.

4.6 Alternative Fairness and Compactness Metrics

Average partisanship. For each district, let its *partisanship* be the percentage of votes in the majority color. Therefore, a low partisanship (towards 50%) implies better competitiveness in that district. We define the average partisanship of Π to be the average of



(a) Average partisanship



(b) Average Polsby-Popper score

Figure 4.6: Distribution of alternative fairness and compactness metrics among subsets of generated plans.

partisanship values over its districts (ignoring small differences in population) as an alternative to the competitiveness metric used in Figure 4.3b. In Figure 4.6a, we compare the average partisanship among the three subsets of plans used in Section 4.5.3 (top-5% fairest plans, whole ensemble, and real enacted plans).

Results show that fair plans generate slightly more partisan districts. However, compared to the whole ensemble, the (roughly) 2-3% of shift in average partisanship is small and comparable to other uncertainties (e.g., voter turnouts or year-to-year election result gaps). Furthermore, the median average partisanship of the fair plans is smaller than that of the real-world redistricting plan for all but one state, showing that local fairness remains compatible with reasonably small partisanship.

Average compactness. We define the average compactness of Π to be the average of Polsby-Popper scores over its districts. In Figure 4.6b we compare the average compactness among the three subsets of plans. Similar to the results for minimum compactness, the average compactness of the locally fair plans remains comparable to that of the entire ensemble. On the other hand, the enacted plans perform better on average compactness than on minimum compactness, showing that enacted plans have larger variances in the compactness scores than plans in the ensemble.

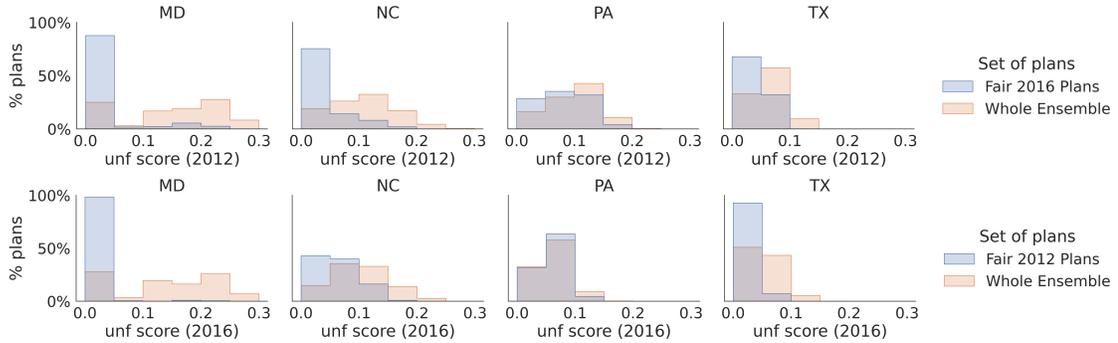


Figure 4.7: Histograms of unf scores. The top plots compute 0.5-fair maps using 2016 voter data, and the blue bars plot the histograms of unf scores when these plans are audited using 2012 voter data. The orange bars represent the histograms of unf scores of the entire ensemble when audited using 2012 data. The bottom plots switch the roles of 2016 and 2012, so the 0.5-fair plans are generated using 2012 data and audited using 2016 data.

4.7 Robustness of Local Fairness to Voting Patterns

To test the robustness of the local fairness notion to changes in voting patterns, we repeat the ensemble-based audit process in Section 4.5 for MD, NC, PA, and TX with $\gamma(v)$, $\tau(v)$, and $\beta(v)$ values replaced by label values obtained from the 2012 presidential election. We do not consider MI (only a few plans are fair, so the sample size is too small) and WI (most plans are fair, and thus the ensemble and fair plans yield similar statistics).

For $c = 0.5$, we obtain the set of locally fair plans (i.e., plans without c -deviating groups) when audited using 2012 (resp. 2016) voter labels. We then compute the unfairness of these plans using the 2016 (resp. 2012) voter data, i.e., from the other election. We repeat this for all the plans in the ensemble, obtaining the unf score for each plan. We plot the histograms of these unf values in Figure 4.7, where the x-axis is the bucketed unf score, and the y-axis is the percentage of plans among the fair maps (resp. ensemble) that fall in that bucket.

For MD, NC, and TX, the blue bars are skewed significantly towards the left compared to the entire ensemble, showing that the fairest plans identified by auditing with a specific election remains significantly fairer compared to the entire ensemble when measured by

another election. This shows the local fairness notion is fairly robust, or insensitive to year-to-year election result fluctuations. For PA, the fair plans are more sensitive to the specific election used, which reflects the role of PA as a swing state across elections.

4.8 Visualization of Fair and Unfair Plans for Other States

We show additional visualizations of fair plans and deviating groups found using ensemble-based auditing. As before, we show deviating groups with black outline, and the districts are coded by its color and the extent of partisanship: districts with a larger value of γ (resp. β) are colored in darker red (resp. blue). For each state (MD, MI, TX, WI), we show a fair plan (Figures 4.8a, 4.9a), and an example of a deviating group of each color. We discuss the deviating groups in each state.

In MD, the central blue districts tend to not be competitive, and the geography of the state contributes to the difficulty of forming red deviating groups. Thus MD is the only state where the precincts in the most deviating groups are not densely populated areas (see Figure 4.2a). Instead, the two “panhandles” (the western and coastal eastern regions of the state) tend to be part of deviating groups, see Figures 4.8b and 4.8c.

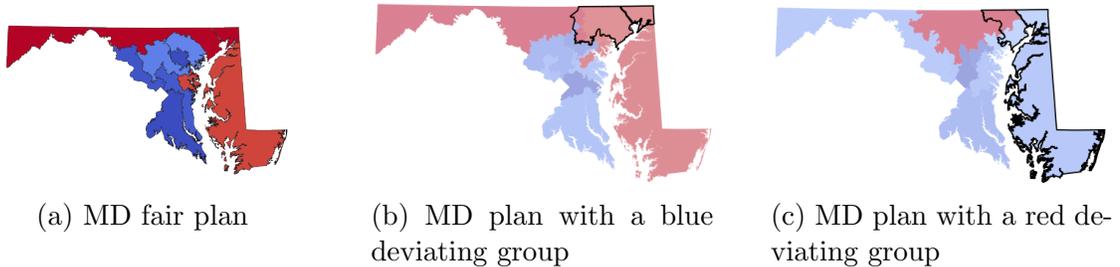


Figure 4.8: Maryland plans without and with deviating groups

In Figures 4.9b and 4.9c we show a red and a blue deviating group in MI. Michigan had the lowest number of fair plans in the ensemble (see Table 4.2). The precincts in deviating groups are clustered in one region of MI, where the districting is sensitive to which precincts belong in red or blue districts. Both red and blue deviating groups are concentrated around

this area.

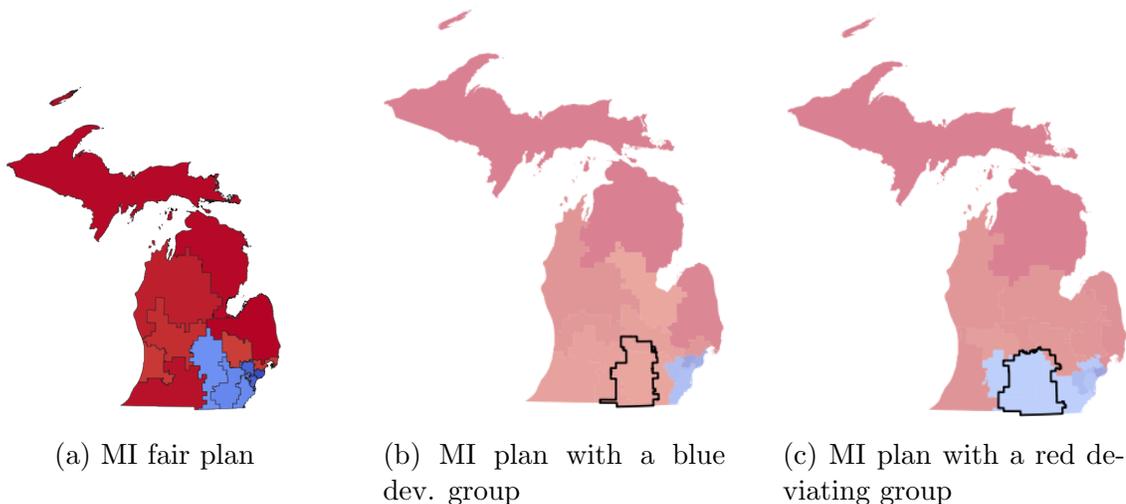


Figure 4.9: Michigan plans without and with deviating groups

4.9 Conclusions

In summary, in contrast to extant global notions of fairness in redistricting such as seat distribution or district competitiveness, the notion of local fairness is an explainable notion that mitigates justified complaints of populations in compact geographic regions. Our experiments show that local fairness is not only possible to achieve in practice, but choosing locally fair plans also does not come at the expense of other global fairness properties.

Several open questions arise from our work. In terms of algorithm design, it is an open question of whether there is an approximation algorithm for either the LF AUDITING or LFP GENERATION problem, and whether such an algorithm could take into account compactness. It would also be interesting to extend our methods to capture additional real-world criteria used in redistricting, such as a penalty for splitting up counties, or a requirement for a majority-minority district. In particular, can fair plans be locally modified so that they remain fair and such real-world criteria are satisfied?

Finally, our exploration of robustness of local fairness to voter turnout is preliminary,

since it compares the outcomes between two election data in one state. It would be interesting to extend our work to a stochastic setting, where each individual in the population has a “likely voter” score (or probability to vote), and we need high confidence in the non-existence of a c -deviating group.

Chapter 5

Redistricting via Random Geometric Partitioning

5.1 Introduction

In this chapter, we continue our study of electoral redistricting in the United States. As mentioned, while redistricting is intended to ensure fair representation, it has a long history of controversy and litigation. One major issue is whether a redistricting plan is “gerrymandered” to favor one political party or demographic group over another. Gerrymandering refers to the manipulation of district boundaries in a way that gives one group of voters an unfair advantage in the elections of a state. This practice has become a contentious issue in American politics, as it can result in one party gaining more seats in government than they would have received under a fair and impartial redistricting plan.

There has been growing interest in approaching the process as an algorithmic problem since the 1960s [BS20]. Various attempts have been made to develop algorithms that can draw district boundaries fairly and impartially, but this remains challenging. Nevertheless, the question of how to design a fair and transparent redistricting process continues to be an active research topic. There is ongoing discussion about what properties a redistricting plan must have. It is commonly agreed that plans should, at minimum, produce population balanced, contiguous, and *compact* districts [Rey64]. However, even with this basic desiderata, exactly how to measure compactness and what constitutes a “reasonably” compact plan is a subject of dispute. Additionally, there is still debate on richer notions of desirability, particularly notions related to the “fairness” of a plan. This has motivated a long line of recent work [DDS21b, DDS20, HRM17] as well as software tools [Pro22, Lev22] on *auditing* a given redistricting plan against various fairness concepts. It should be noted that under most notions of desirability proposed in the literature, the problem of redistricting

is computationally hard [KMV19], leading to the study of heuristic approaches. We have investigated these fairness notions in the previous chapter, and we now delve deeper the ReCom algorithm and other Markov chain approaches to redistricting

Markov chain approaches. More recently, there has been a focus on generating *ensembles* of redistricting plans¹, which are then used to *audit* a redistricting plan. For any measure of interest, such as compactness or partisanship, the goal is to sample redistricting plans to form a sufficiently rich ensemble over the set of all plans, which can capture the properties of the distribution on the property of interest. Most methods are non-partisan, in that they only take into account geographic information on individuals – that is, state geography and population distribution – while generating the ensemble, and this ensemble is subsequently used to compare a plan being audited on their partisan and non-partisan properties. A natural approach for ensemble generation is Markov Chain Monte Carlo (MCMC). In this work, focus on the *recombination* MCMC method (called ReCom) because it is a widely adopted tool for generating ensembles, and is also among the most efficient of such MCMC methods. In this method, at each step, a pair of adjacent districts is merged, a random spanning tree is computed on this merged district, and this tree is split into two population balanced parts. The recombination chain can produce a large set of different redistricting plans along with explanation of how various properties on the plans are distributed [DT18]. We note that such an ensemble approach can not only be used for auditing, but also for generating a plan itself, since the planners can examine the plans in a small ensemble and choose one that is desirable on their metrics of choice.

One drawback of ReCom is its viewing of redistricting as a graph partitioning problem: Its input is a planar graph representation of the state where vertices are precincts, and roughly speaking, it views the redistricting as finding a random spanning tree and partitioning this tree into the right number of population balanced districts.² While there is some evidence that this biases the chain towards compact districts [PTF22], in practice, the

¹Recall, an ensemble is a set of outcomes produced by an iterated random sampling process such as a Markov chain.

²This is not strictly speaking true; see [CLLV22] for a discussion.

districts produced don't "look" compact. In other words, the outcome of ReCom appears closer to the ensemble of *all possible* redistricting plans, with some bias towards compact plans. Though one could bias the chain much more towards producing more compact districts, this approach is not principled and may not preserve other desirable properties such as seat share.

In this chapter, we ask: Is there an alternate view of the redistricting problem where a naturally defined Markov Chain produces much more compact plans, while not only preserving all other desirable properties of ReCom such as computational efficiency, but also generating an ensemble that is "close to" the (roughly) all-encompassing ensemble produced by ReCom? *Such a chain would yield a compact analog of sampling a random redistricting plan.* We answer this question in the affirmative by viewing redistricting as a geometric problem instead of a graph problem.

5.1.1 Our Contribution

In this chapter, we introduce a new family of Markov chains for redistricting, which we call Geometric ReCom, by modifying the standard ReCom chain. As with ReCom, Geometric ReCom merges and splits neighboring districts in each iteration. However, we adopt a geometric approach in the split step by fitting an ellipse on the combined district, and picking a random direction on the ellipse to cut the combined district into two. We empirically show that compared to ReCom, Geometric ReCom produces a smaller ensemble of redistricting plans that are significantly more compact under multiple measures of compactness. Surprisingly, we show that though this ensemble is much smaller, it preserves the distribution of other desirable properties such as seat shares and competitiveness. Further, this ensemble achieves good coverage of the ReCom ensemble in the sense that for any plan in the latter ensemble, there is a plan in the former ensemble that is close under a natural notion of closeness. Therefore, it can be viewed as a compact analog of sampling a random redistricting plan.

In Section 5.3, we present Geometric ReCom. We illustrate how Geometric ReCom works

on a 6×6 grid and show that Geometric ReCom mixes quickly on this toy example. We show that under a natural initial partition, ReCom generates all possible partitions on this example, while Geometric ReCom is much more focused, generating only around 40% of the partitions, and these are significantly more compact compared to the average partition generated by ReCom. Nevertheless, we show that the Geometric ReCom ensemble is dense in the set of all partitions in some sense.

In Section 5.4, we empirically compare Geometric ReCom with ReCom on data from real-world elections. We again show that Geometric ReCom produces significantly more compact plans compared to ReCom, and these plans are comparable in metrics such as seat share outcomes and the number of competitive districts. Though Geometric ReCom generates a different and possibly smaller ensemble, we empirically show in Section 5.4.2 that for any plan in the ReCom ensemble, there is a plan in the Geometric ReCom ensemble that is similar enough, which provides some evidence that Geometric ReCom is sampling a random compact redistricting plan.

5.1.2 Related Work

The utilization of computational tools in the field of redistricting has a longstanding history, tracing back to the 1960s [Vic61, HWS⁺65]. Over the years, extensive research (refer to [BS20] for a comprehensive survey) cast the task as an optimization problem, in which the goal is to generate a feasible redistricting plan. The objective is to generate a feasible redistricting plan that satisfies crucial criteria such as population balance, connectedness, and compactness, while solely relying on spatial information pertaining to individuals within a state. This problem is computationally intractable in the worst case [CKM⁺21, KMV19]. Nonetheless, numerous techniques have been proposed to address this challenge, including the use of Voronoi diagrams [LF19, GKM⁺21, CAKY18], local search algorithms [KJS15], simulated-annealing [AM11], and evolutionary algorithms [LCW16b]. Like the proposed method, the Voronoi diagram approach to redistricting has a similar geometric flavor. However, these methods focus on computing a single optimal redistricting plan, rather

than sampling from the space of feasible plans.

In addition to generating feasible redistricting plans, there is also much interest in *auditing* a proposed plan, including in court cases and for state legislatures [MKS⁺22, Mat21]. The goal of auditing is to determine whether a redistricting plan has been gerrymandered. Several approaches to auditing have been proposed, such as relying on appropriate desirability scores. These are typically scores assigned to a redistricting plan and can be based on a number of properties, such as compactness, including the Reock [Reo61] and Polsby-Popper [PP91] scores. Other scores take into account electoral outcomes, and include the efficiency gap [SM15], mean-median gap [Wan16], partisan symmetry [War18], the GEO metric [CRSV22], and the competitiveness of the plan [DDS20]. Many of these measures are used in publicly available tools for current enacted districtings [Pro22, CRSV22]. Other work on auditing proposes new fairness metrics [KTAM22] or methods for combating gerrymandering, such as a recent work that attempts to show the effects of gerrymandering can be tempered by replacing single-member district winners with multiwinner elections [GGRS22].

Since the 2010s, a popular line of work has focused on generating *ensembles* of redistricting plans for use in auditing. An ensemble is a diverse set of redistricting plans, usually containing thousands of plans, that is a representative sample of the space of desirable redistricting plans. In this space, Chen and Rodden provided the first example in which randomized plan generation would be used to quantify the bias in a proposed or enacted plan [CR13]. Other methods include Flood Fill [CDO00, MM18], Column Generation [GS21], and the widely adopted Markov Chain Monte Carlo (MCMC) approach [TCL16, FHIT20, LCC⁺22]. It is shown in [PTF22] that the widely used MCMC method [DDS21b] provably biases towards compact plans, we provide a more in-depth description of ReCom in Section X. Ensemble based approaches provide a natural, statistical way of auditing [HRM17, HKL⁺20]: The properties of the enacted plan is compared against the distribution of the corresponding property on the ensemble; if the plan is a statistical outlier, then it is considered more “gerrymandered” and hence less desirable. New uses for

ensembles of plans have also emerged. The recent work of [LCC⁺22] instead uses plans in the ensemble as comparators to identify manipulation in redistricting plans. The work of [EGB22] proposes a method for choosing one representative plan from such an ensemble based on defining distances between plans. There is also work to produce ensembles that are publicly available for use in auditing [MKS⁺22].

5.2 Preliminaries

The input consists of a precinct map M of a state, which is a polygonal planar subdivision, each of whose face is a polygon called a precinct. Let $G = (V, E)$ be the planar dual graph of M , where each vertex $v \in V$ represents a precinct of M , and there is an edge between two vertices if and only if the corresponding precincts are geographically adjacent. We refer to G as the *precinct graph*.

For each precinct $v \in V$, let $\rho(v) > 0$ denote its population and let $\tau(v) \in [0, \rho(v)]$ denote the number of voters in v . Let $\gamma(v) \in [0, 1]$ and $\beta(v) = 1 - \gamma(v)$ denote the fraction of $\tau(v)$ who vote *red* and *blue*, respectively. It is implicitly assumed that each voter is exactly one of the two colors. For an arbitrary set of precincts $W \subseteq V$, let $\rho(W) = \sum_{v \in W} \rho(v)$, the total population of precincts in W .

Definition 36 *k*-redistricting plan. A *k*-redistricting plan Π of $G = (V, E)$ is a partition of V into *k* pairwise-disjoint subsets (or districts) $D_1, D_2, \dots, D_k \subseteq V$.

Given *k*, desired number of districts and an error parameter $\varepsilon \in (0, 1)$, we say a district $D \subseteq V$ is ε -feasible if

- (i) D induces a connected subgraph and
- (ii) the population of D is at most ε away from average, *i.e.*,

$$(1 - \varepsilon) \frac{\rho(V)}{k} \leq \rho(D) \leq (1 + \varepsilon) \frac{\rho(V)}{k}.$$

A redistricting plan Π is ε -feasible if each district $D_i \in \Pi$ is ε -feasible. Note that this definition is consistent with the general practice in the U.S. where the population of a precinct v is determined not by eligible voters $\tau(v)$, but by the total population $\rho(v)$.

Compactness. In addition to requiring that districts be contiguous and population balanced, many redistricting models require that the districts be “compact” as well. In real redistricting tasks, a non-compact district not only makes less geographic sense, but is also more likely to have been gerrymandered to favor one party over another and is one important measure to evaluate proposed redistricting plans in US states. There is no hard constraint on the definition of compactness as it can be defined both in terms of the geometry of districts and the edges of the precinct graph. Intuitively, a district is compact if the ratio of its perimeter and area is small (geometric property) or has few “cut” edges whose endpoints are in different districts (graph property). There are widely accepted measures of compactness, some of which have been used by the courts to rule on whether a redistricting plan is legal [Mil95, BH17]. In our work, we consider the following two measures of compactness:

- (i) *Polsby-Popper score*: The Polsby-Popper score of a district D is $\frac{4\pi A(D)}{P(D)^2}$, where $P(D)$ is the perimeter of the district D , and $A(D)$ is the area of the district D . A higher score indicates a district is more compact. This score is also used to evaluate the compactness of plans in practice [BH17, Mat21].
- (ii) *Percentage of Cut Edges*: For a redistricting plan Π , we count the number of edges whose endpoints lie in different districts of Π : the size of $C = \{e = (u, v) : u \in D_j, v \in D_i, i \neq j\}$. Intuitively, more compact redistricting plans have fewer number of cut edges. Let the cut edge measure of compactness be defined as $|C|/|E|$.

Similarity and Coverage. Given two districting plans Π_1 and Π_2 , let the *similarity matrix* S be a $k \times k$ matrix S where S_{ij} is the number of precincts assigned both to district $D_i \in \Pi_1$ and to district $D_j \in \Pi_2$. Note that if $\Pi_1 = \Pi_2$, there will be exactly k non-zero

entries in S . Let the *similarity* of Π_1 and Π_2 be the sum of the largest k entries in S over the total number of precincts: $|V|$, and denote this score by $sim(\Pi_1, \Pi_2)$.

We use this similarity measure to evaluate *coverage*. Let the set of all plans generated by ReCom be $\mathcal{P} = \{\Pi_1, \dots, \Pi_n\}$. We compute $sim(\Pi_i, \Pi_j)$ for all $i \neq j$. Let the scores computed be S . Let the plans generated by Geometric ReCom be $\mathcal{G} = \{\Gamma_1, \dots, \Gamma_m\}$. For each plan $\Pi \in \mathcal{P}$, we compute the plan $\Gamma \in \mathcal{G}$ that is the most similar to Π , as measured by $sim(\Pi, \Gamma)$ and let these matching scores be denoted by M . To see how well the set of plans of Geometric ReCom covers the same space of plans of ReCom, for each plan $\Pi \in \mathcal{P}$ we compute the percentile rank of scores in M with respect to S . If there is a large fraction of M with a high percentile rank, this indicates that for most plans of ReCom, Geometric ReCom can generate a similar plan.

5.3 A New Family of Markov Chains

ReCom Overview. The family of Markov chains known as recombination is widely used in both theoretical and applied work related to redistricting, and has been used in the auditing process of real redistricting proposals [DDS21b, BS20]. The ReCom chain is a large-step random walk in the space of partitions. The “large” step changes multiple districts’ compositions at a time. The state space is the full set of (ε) -feasible (k) -partitions of a precinct graph G . In each step, ReCom randomly combines two (or more in its general form) districts of the current redistricting plan, generates a random spanning tree of the subgraph induced by the combined districts, and re-partitions the subgraph into two parts by cutting edges in the spanning tree so that the new resulting districts remain ε -feasible.

This approximately targets the spanning tree distribution on plans, i.e., the probability it generates a given plan depends on the spanning-tree compactness of the plan as measured by the product over the districts in the plan of the number of spanning trees in that district. It is shown that the random spanning tree step *biases* the Markov chain towards plans with compact districts, where compactness is measured by number of cut edges [PTF22].

However, ReCom may produce unrealistic redistricting plans that are too non-compact when viewed as geometric regions rather than a planar graph.

In the following, we present a new family of Markov Chains, which we call Geometric ReCom, that produce redistricting plans which are more compact than those produced by ReCom, while achieving similar distributions in other aspects (such as seat share percentages of electoral parties).

5.3.1 Geometric ReCom Proposal

In this section, we present the detailed procedure for Geometric ReCom. The algorithm is similar to ReCom at a high level, but adopts a geometric approach to split districts at each step. As with ReCom, the basic splitting procedure will not consider demographic or election information other than precinct population. For a vertex $v \in V$ in the precinct graph, we choose a point $p_v \in \mathbb{R}^2$ that lies in the precinct represented by v (in the precinct map), say, the centroid of the precinct (if it indeed lies within the precinct).

At each step of the chain, we have a map Π . We randomly combine two adjacent districts, D_i and D_j , of Π and then re-split $\bar{D} = D_i \cup D_j$ into two districts D'_i, D'_j . $\Pi' = (\Pi \setminus \{D_i, D_j\})$ is a new map. We describe the splitting procedure that splits \bar{D} into D'_i, D'_j . For a district $D_k \in \Pi$, let $P_k = \{p_v \mid v \in D_k\}$. Set $\bar{P} = P_i \cup P_j$ to be the set of points corresponding to the precincts in \bar{D} . For a line $\ell \in \mathbb{R}^2$, let $V_L(\ell) \subseteq \bar{D}$ (resp. $V_R(\ell) \subseteq \bar{D}$) be the set of precincts v such that p_v lies to the left (resp. right) of ℓ . First, we fit an ellipse \mathcal{E} through \bar{P} based on least-squares fitting [GGS94]. In particular, the sum of the squares of the distances of the points of \bar{P} from $\partial\mathcal{E}$, the boundary of \mathcal{E} , is minimum. Let x_o be the center of \mathcal{E} . We randomly pick a point q on $\partial\mathcal{E}$. We project \bar{P} onto the line g containing the segment x_oq . For a point $x \in g$, let ℓ_x be the line passing through x and orthogonal to g . Let $I = \{x \in g \mid V_L(\ell_x), V_R(\ell_x) \text{ are } \varepsilon\text{-feasible}\}$. We choose a random point $a \in I$. If $V_L(\ell_a), V_R(\ell_a)$ form connected subgraphs, i.e., the corresponding regions in the precinct map M are connected, we set $D'_i = V_L(\ell_a)$ and $D'_j = V_R(\ell_a)$. We give an example of a feasible cut in Figure 5.1 The detailed algorithm for this update step is presented in

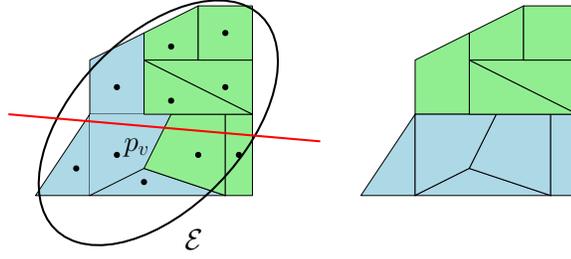


Figure 5.1: For our initial plan Π on the right, the set of precincts \bar{P} are the black points and the ellipse is \mathcal{E} . After choosing the red line as ℓ_a , we obtain the partition on the right as our new plan Π' .

Algorithm 5.2.

Compactness. Here is the intuition on why our splitting procedure leads to more compact maps. First, the ellipse \mathcal{E} is an approximation of the polygon $D_i \cup D_j$. By choosing a random point q on $\partial\mathcal{E}$ and partitioning \mathcal{E} into two parts by a line orthogonal to x_oq , we partition \mathcal{E} into two parts by a chord whose expected length is not much larger than the length of its minor axis. If the precincts of M are compact then the boundary line of D'_i and D'_j is roughly the same as the length of the chord that generates the splitting of \bar{D} into D'_i and D'_j . In other words, our procedure biases the split of \bar{D} into two districts D'_i, D'_j such that their common boundary has small length. Choosing this point q is equivalent to transforming the ellipse to a circle and choosing a random direction. We show an example of a plan found by ReCom and a plan found by Geometric ReCom in Figure 5.3 in two different states (Wisconsin and Pennsylvania). For this figure, we show a ReCom plan and a similar map found by Geometric ReCom. Note that the boundaries of districts generated by Geometric ReCom are shorter and thus the districts are more compact (see e.g., the purple and orange districts in Figure 5.3c and the corresponding districts in Figure 5.3d).

Coverage. Although there are exponential number of ways to partition \bar{P} , and thus \bar{D} , we consider only $O(|\bar{P}|^2)$ of them since that is the number of distinct partitions by a line. Since we partition \bar{P} by a line, the boundary of D'_i and D'_j is much smoother, as evident from Figure 5.3. The goal of computing an ensemble is to *sample* in the space of desirable

Geometric ReCom:

input: $G = (V, E)$, the current partition $\Pi = \{D_i, \dots, D_k\}$, population parameter ε
 Select two distinct adjacent districts D_i, D_j from Π uniformly
 Let $\bar{D} = D_i \cup D_j$ and $\bar{P} = P_i \cup P_j$
 $\mathcal{E} \leftarrow$ Least-squares ellipse with center x_o on \bar{P}
 $q \leftarrow$ uniform random point on $\partial\mathcal{E}$
 Let g be line containing segment x_oq
 Compute $I = \{x \in g \mid V_L(\ell_x), V_R(\ell_x) \text{ are } \varepsilon\text{-feasible}\}$
 Pick $a \in I$ uniformly at random
if the subgraph of G induced by $V_L(\ell_a)$ or $V_R(\ell_a)$ is not connected:
 return: $\Pi' = \Pi$
else:
 return: $\Pi' = \Pi \setminus \{D_i, D_j\} \cup \{V_L(\ell_a), V_R(\ell_a)\}$

Figure 5.2: Geometric ReCom

plans (those which are ε -feasible and compact). We do not want to narrow the space of plans we sample from too much compared to ReCom, we discuss this more in Section 5.3.2 and 5.4.2.

5.3.2 Behavior of Markov Chain on Simple Graphs

Before presenting our main empirical results in Section 5.4, we first present results on a simple example. We show that Geometric ReCom not only mixes quickly on a grid graph, but also finds more compact partitions. The input is a 6×6 grid, with each square representing a precinct. We want to divide the grid into 4 districts of equal size, so that each district contains 9 squares. Figure 5.4 presents a schematic of a Geometric ReCom recombination step on this example.

There are roughly 20,000 ways to partition a 6×6 grid into 4 districts so that each district is connected if we count isomorphic partitions as a single partition. Using computer-assisted search, we find that starting with any partition as the initial partition and running Geometric ReCom, we can reach the *most* compact partition (the leftmost partition in Figure 5.4) within 1,000 steps of the chain with probability higher than 90%. This implies that Geometric ReCom mixes quickly on this example.

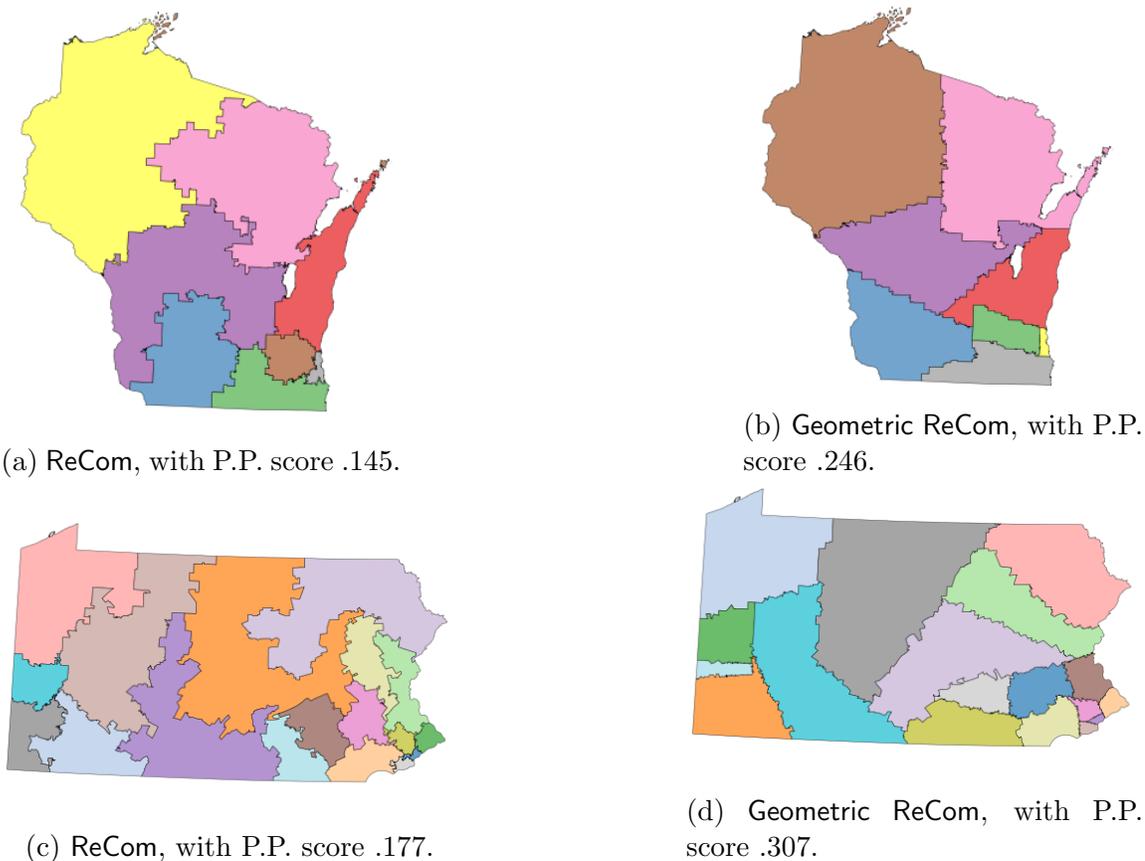


Figure 5.3: Comparison of Plans generated by algorithms, the average Polsby-Popper score is shown.

Producing Compact Partitions. We demonstrate that Geometric ReCom produces more compact partitions than ReCom on this example. We use the most compact partition (the leftmost partition in Figure 5.4) as our initial partition, as any initial partition can reach this partition in a small number of steps with high probability. We run ReCom and Geometric ReCom for 100,000 iterations, respectively, and compare the compactness of the partitions produced by these two algorithms. Figure 5.5a (resp. Figure 5.5b) presents the distribution of the average Polsby-Popper score of the four districts (resp. the percentage of cut edges) for both ReCom and Geometric ReCom. In Figure 5.5a, higher x -axis value is better; in Figure 5.5b, lower x -axis value is better. We see that when compactness is measured in terms of either criteria, Geometric ReCom tends to produce more compact partitions.

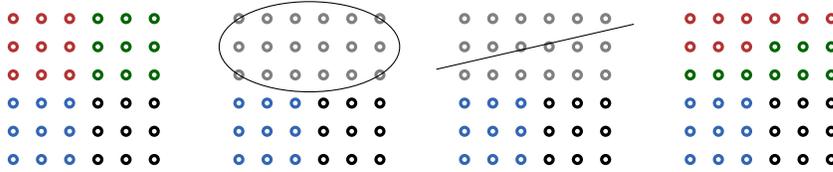


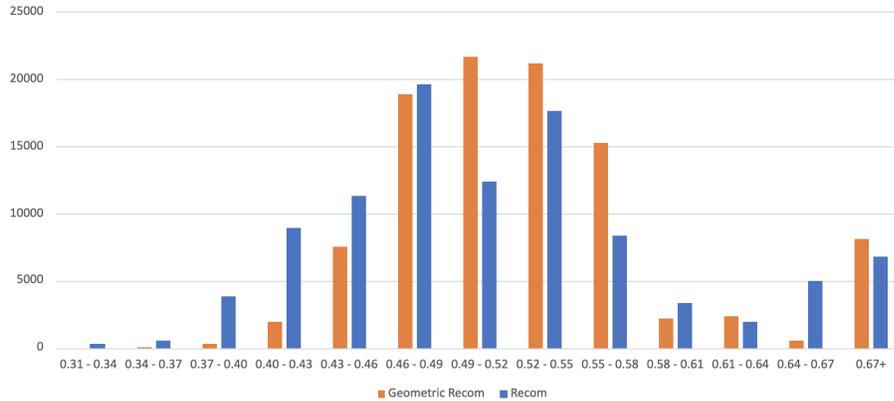
Figure 5.4: Illustration of a Geometric ReCom step for a 6×6 grid with 4 districts.

Coverage of Geometric ReCom. Using computer-assisted search, we find that: If we use the most compact partition as our initial partition, ReCom could generate all possible partitions, while Geometric ReCom generates around 40% of the partitions. However, as we have seen, Geometric ReCom tends to focus on more compact partitions. Moreover, we show that for any possible partition, there exists a partition that Geometric ReCom produces and is similar to this partition by computing similarity scores between ReCom plans S and scores between ReCom plans and their best matching Geometric ReCom plan M , as described in Section 5.2. We find that for all $\Pi \in \mathcal{P}$, $\text{sim}(\Pi, \sigma(\Pi))$ is above the 80th percentile in S ; for over 95% of $\Pi \in \mathcal{P}$, $\text{sim}(\Pi, \sigma(\Pi))$ is above the 90th percentile, and over 90% is above the 95th percentile. This implies that the Geometric ReCom ensemble nicely covers the set of all possible partitions.

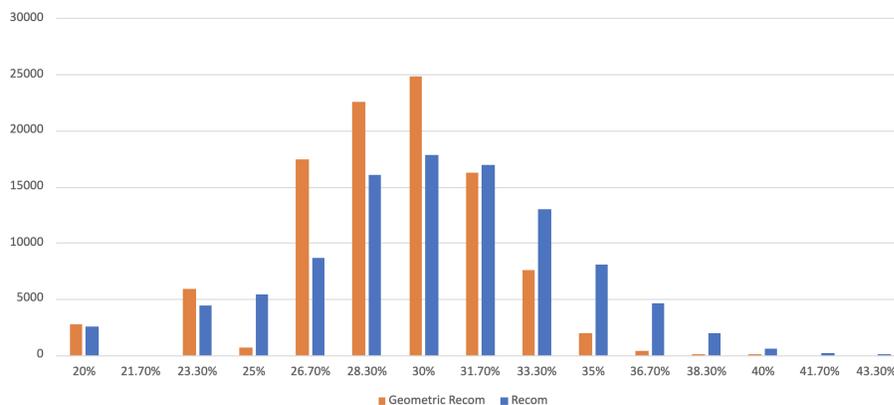
5.4 Experiments

As discussed, one goal of ReCom is to sample the space of desirable redistricting plans. If the set of plans generated will be used to audit a real plan, it is important that we have sufficient coverage of the space of compact feasible plans, otherwise we may label a good plan an outlier compared to our ensemble or label an undesirable plan “good” if we miss sampling from large parts of the state space.

In our experiments, we attempt to answer the following questions. First, does using Geometric ReCom compute a set of more compact redistricting plans than ReCom? Is the resulting set of plans suitably diverse, or have coverage similar to ReCom? Does it produce comparable distributions on other measures of interest?



(a) Polsby-Popper score comparison.



(b) Percentage of cut edges comparison.

Figure 5.5: Comparison between ReCom and Geometric ReCom on the simple example. The horizontal axis is the average Polsby-Popper score (resp. percentage of cut edges) and the vertical axis is the number of plans with that Polsby-Popper score (resp. percentage of cut edges) value.

5.4.1 Datasets and Methods

Data used in our experiments is obtained from the Redistricting Data Hub and MGGG states [MGG22, Hub23]. We obtain precinct graphs and voter information for Maryland (MD), Michigan (MI), North Carolina (NC), Pennsylvania (PA), Texas (TX). We set $\rho(v)$ to the 2010 census population in each precinct v . The default election we use is the 2016 congressional election. For each precinct v , we collect the number of total votes for the Republican party r_v and the total vote amount for the Democrat party b_v in the election. We set $\gamma(v) = r_v / (r_v + b_v)$, $\beta(v) = 1 - r_v$, and the total number of votes $\tau(v) = r_v + b_v$.

We generate two ensembles of redistricting plans: one by running ReCom and the other running Geometric ReCom. Each ensemble consists of 1,000 redistricting plans, each being the outcome of a 10,000-step Markov chain with population balance parameter $\varepsilon = 0.02$. We seed each chain with a recent congressional electoral districting plan of the state. We set k , the number of districts, to be the number of congressional districts in the 2016 election for each state.

Compactness We compare the average and minimum Polsby-Popper scores of the redistricting plans generated by both Geometric ReCom and ReCom, see Figure 5.6a and 5.6b. For each state, we show a split violin plot: on the left in the lighter color is the distribution of ReCom (denoted by ‘R’ in the legend), and on the right in the darker color is the distribution of Geometric ReCom (denoted by ‘G’). For all states Geometric ReCom produces more compact plans than ReCom. The minimum Polsby-Popper scores tend to be more similar between the two algorithms, but this could be due to geographic features of the state.

We also compare the number of cut edges generated by the two algorithms as a percentage of total edges in the graph. For this measure, a lower percentage intuitively implies a more compact district. The percentage of cut edges in a plan are lower for the ensemble of plans generated by Geometric ReCom, though to a lesser difference than in the Polsby-Popper score. This is somewhat surprising, as Geometric ReCom does not use a graph-based splitting method like ReCom.

Other Metrics We compare the set of plans generated by the two algorithms and measure two properties of interest: the seat outcome and the competitiveness of a plan. While we want to produce more compact plans, we do not want the cost of such plans to limit the possible distribution on other measures of interest. Our main finding is that Geometric ReCom and ReCom find similar distributions of each property.

The resulting distribution is shown in Figure 5.7a. The Princeton Gerrymandering Project [Pro22] defines a district as *competitive* if the majority color is at most 53.5% of

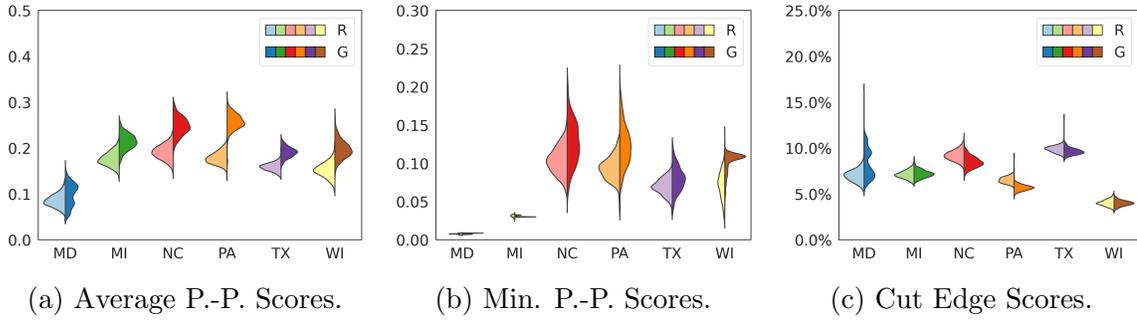


Figure 5.6: Compactness comparison between Geometric ReCom and ReCom. On the horizontal axis is the state, on the vertical axis is the score distribution.

the total votes. We count the number of such districts, and compute the percentage of competitive districts. We show the distribution of the percentage of competitive districts in Figure 5.7b.

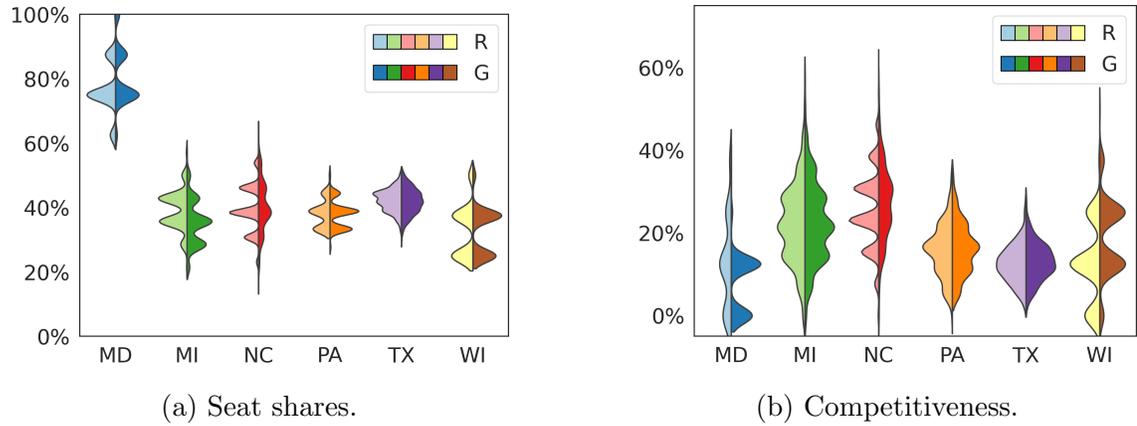
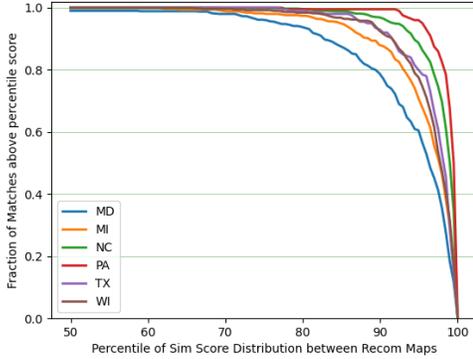


Figure 5.7: Other metrics of interest comparison between Geometric ReCom and ReCom. In (a), seat share is represented as percent of districts with a blue majority. In (b), we plot the percent of districts being competitive (having less than 53.5% voters in majority).

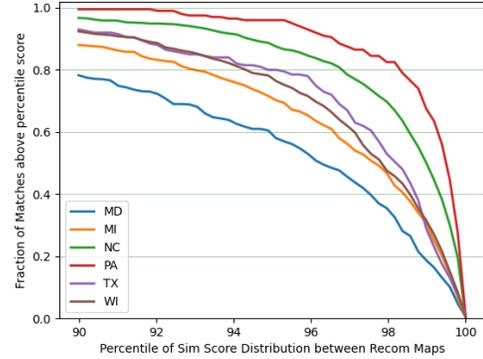
5.4.2 Sampling Space

We next consider whether the plans generated by Geometric ReCom are *similar* to plans generated by ReCom, our goal is to show that we achieve good *coverage* of relevant redistricting plans. We compute sets S and M as described in Section 5.2. For each plan $\Pi \in \mathcal{P}$,

we compute the percentile rank of $\text{sim}(\Pi, \sigma(\Pi))$ with respect to S . We plot the fraction of scores in M above a percentile, see Figure 5.8a. In Figure 5.8, we show the graph for the 90th-100th percentile. As shown, for all states, there is a large fraction of scores in M above the 90th percentile. This indicates that for each ReCom plan, a similar Geometric ReCom plan can also be generated.



(a) Matches above x^{th} percentile.



(b) Breakdown of the 90th–100th percentile.

Figure 5.8: Coverage of ReCom by Geometric ReCom and. On the horizontal axis is the percentile in S and on the vertical axis is the fraction of matches above the corresponding percentile.

5.5 Conclusion

In this chapter, we presented a novel Markov Chain Monte Carlo (MCMC) method for electoral redistricting which aims to generate redistricting plan ensembles. By adapting the widely used recombination chain and incorporating geometric partitioning, our method outperforms previous approaches in terms of compactness scores, both through graph cuts and geometric properties, while maintaining competitiveness and similar seat share outcomes. Despite generating plans from a smaller space, our method demonstrates good coverage, ensuring the ensemble contains plans close to any desirable configuration. Future work could further explore the potential of the proposed method, including how to incorporate additional constraints required for real redistricting applications and further describe the

space of plans the chain finds.

Chapter 6

Multi-Robot Motion Planning for Unit Discs with Revolving Areas

6.1 Introduction

Multi-robot systems are already in use in logistics, in a variety of civil engineering and nature preserving tasks, and in agriculture, to name a few areas. They are anticipated to proliferate in the coming years, and accordingly they attract intensive research efforts in diverse communities.

A basic motion-planning problem for a team of robots is to plan a collision-free paths for the robots between given start and final positions. Among the many dimensions along which the multi-robot motion planning (MRMP) problem has been studied, we focus on three: (1) we distinguish between distributed and centralized control. In the former each robot has limited knowledge of the entire environment where the robots move, and each robot may communicate with few neighboring robots. In the latter, which is typical in factory automation and other well-structured environments, a central authority has control over all the robots and the planning for each robot takes into consideration knowledge about the state of all the other robots in the system. (2) In the *labeled* version the robots are distinguishable from one another and each robot has its own assigned target, whereas in the *unlabeled* version the robots are indistinguishable, i.e., each target can be occupied by any robot in the team and the motion-planning problem is considered solved if at the end of the motion all the target positions are occupied. (3) We further distinguish between *continuous* or *discrete* domains. Much of the study of motion planning in computational geometry and robotics assumes that the workspace is *continuous*. In AI research, where the problem is typically called multi-agent path finding (MAPF) [SSF⁺19], the domain is modeled as a graph. Nowadays the MAPF problem is studied in diverse research communities, often as

an approximation of the continuous domain.

In our study here we consider a *centralized, labeled, and continuous* version of MRMP. Furthermore, we are not only interested in finding a solution to the given motion-planning problem, but rather in finding a high-quality solution. Specifically, we aim to find a solution that minimizes the total path length traveled by the robots.

Related Work. Computing a feasible motion plan (not necessarily a good one) itself is in general computationally hard for MRMP (see, e.g., [HSS84, BCD⁺21, SH16, GH21]). In the results that we cite next, some additional mitigating conditions are assumed on the system to obtain efficient motion-planning algorithms.

There are few results that guarantee bounds on the quality of the motion plans for multi-robot systems. For complete algorithms¹ in the unlabeled case, there are bounds on the length of the longest path taken by a robot in the system [TMMK14], or on the sum of distance traveled by all the robots [SYZH15]. For the labeled case, Demaine et al. [DFK⁺19] provide constant-factor approximation algorithms for minimizing the execution time of a coordinated parallel motion if there are no obstacles. Still for the labeled case, Solomon and Halperin obtained a very crude bound on the sum of distances [SH18] (the approximation factor can be linear in the complexity of the environment in the worst case) in a setting identical to the setting of the current chapter, namely assuming the existence of *revolving areas*—see below for a formal definition. No sublinear approximation algorithm is known for MRMP even if we assume the existence of revolving areas and the cost of a motion plan is the sum of the lengths of individual paths. In the current work we significantly improve over and expand the results in [SH18] in several ways, as we discuss below.

An alternative approach to cope with the hardness of motion planning is to use *sampling-based* methods [Sal19]. In their seminal paper, Karaman and Frazzoli [KF11] (see also [SJS⁺20]) introduced an algorithm, called RRT*, which guarantees near optimality if the number of samples tends to infinity. A related algorithm dRRT* handles the multi-robot case with

¹A motion planning algorithm is called *complete* if, in finite time, it is guaranteed to find a solution or determine that no solution exists.

the same type of guarantee [SSD⁺20]. Recently Dayan et al. [DSPH21] have obtained near-optimality with finite sample size for the multi-robot case.

Problem Statement. Let \mathcal{W} be a polygonal environment, that is, a polygon with holes in \mathbb{R}^2 and a total of m vertices. Let R_1, \dots, R_n be n robots, each modeled as a unit disc, that move around in \mathcal{W} . Let $\mathcal{O} = \mathbb{R}^2 \setminus \mathcal{W}$ be the obstacle space. For a point $p \in \mathbb{R}^2$, let D_p denote the unit disc centered at point p . Let $\mathcal{F} = \{x \in \mathcal{W} : D_x \cap \mathcal{O} = \emptyset\}$ represent the free space of \mathcal{W} (with respect to one R_i). A *path* is a continuous function $\pi : I \rightarrow \mathbb{R}^2$ from an interval I to \mathbb{R}^2 , and is *collision-free* if it is contained in \mathcal{F} . Let $\ell(\pi)$ denote the arc length of π , i.e., $\ell(\pi) = \int_I |\pi'(t)| dt$. The position of each R_i is specified by the x - and y -coordinates of its center c_i and we use $R_i(c)$ to denote R_i being at c (note that $R_i(c)$ is the same as D_c), and a motion of R_i is specified by the path followed by its center. Let $\text{int } D$ denote the interior of disc D . A *path ensemble* $\Pi = \{\pi_1, \dots, \pi_n\}$ is a set of n paths defined over a common interval I , i.e. $\pi_i : I \rightarrow \mathbb{R}^2$, for $1 \leq i \leq n$; Π is called *feasible* if (i) $\pi_i \subset \mathcal{F}$ for every $i \leq n$, and (ii) for any $t \in I$ and for any pair $i \neq j$, $\text{int } R_i(\pi_i(t)) \cap \text{int } R_j(\pi_j(t)) = \emptyset$, i.e., the R_i 's remain in \mathcal{W} and they do not *collide* with each other (but may touch each other) during the entire motion. We also refer to Π as a *motion plan* of R_1, \dots, R_n . The cost of Π , denoted by $\mathcal{C}(\Pi)$, is defined as $\mathcal{C}(\Pi) = \sum_{i=1}^n \ell(\pi_i)$.

We are given a set of *start* positions $\mathbf{s} = \{s_1, \dots, s_n\}$ where the n robots initially lie and a set of *final* (also called *target*) positions $\mathbf{f} = \{f_1, \dots, f_n\}$. Our goal is to find a path ensemble $\Pi^* = \{\pi_1^*, \dots, \pi_n^*\}$ over an interval $[0, T]$ where T denotes the ending time of the last robot movement,

(i) $\pi_i^*(0) = s_i$ and $\pi_i^*(T) = f_i$ for all i , and

(ii) $\mathcal{C}(\Pi^*) = \min_{\Pi} \mathcal{C}(\Pi)$ where the minimum is taken over all feasible path ensembles.

We refer to the problem as *optimal multi-robot motion planning (MRMP)*. In this chapter, we investigate optimal MRMP under the assumption that there is some free space around the starting and final positions of R_1, \dots, R_n , a formulation introduced in [SH18]. A *revolving area* of a start or final position $z \in \mathbf{s} \cup \mathbf{f}$, is a disc A_z of radius 2 such that:

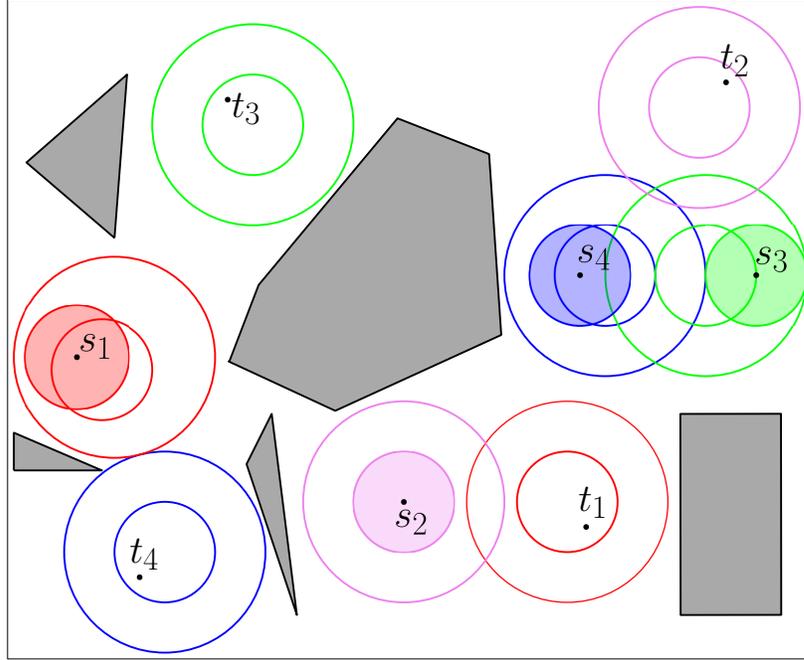


Figure 6.1: We show an instance of MRMP-RA. Each robot is shown as a filled disc in its starting revolving area, and its target revolving area is shown in the same color. Obstacles are dark gray.

1. $D_z \subseteq A_z$,
2. $A_z \cap \mathcal{O} = \emptyset$, and
3. for any other start or final position $y \in (\mathbf{s} \cup \mathbf{f}) \setminus \{z\}$, $A_z \cap D_y = \emptyset$.

That is, each R_i lies in a revolving area at its start and final position (note that z need not be the center of the revolving area A_z) and does not intersect any other revolving areas, and the revolving areas do not intersect any obstacles. We remark that the revolving areas may intersect one another; this makes the separation assumptions in the current work lighter than in related results (e.g.,[AdBHS15]), which in turn makes the analysis more involved. See Figure 6.1 for an example. Set $\mathcal{A} = \{A_z : z \in \mathbf{s} \cup \mathbf{f}\}$. We refer to this problem as *optimal multi-robot motion planning with Revolving Areas (MRMP-RA)*.

We define the *active interval* $\tau_i \subseteq [0, 1]$ as the open interval from the first time R_i leaves the revolving area A_{s_i} of s_i to the last time R_i is not in the revolving area A_{f_i} of f_i . If the active intervals $\{\tau_1, \dots, \tau_n\}$ are pairwise disjoint then we call Π a *weakly-monotone motion*

plan (with respect to revolving areas).² Finally, an instance of optimal MRMP is specified as $\mathcal{J} = (\mathcal{W}, \mathbf{s}, \mathbf{f})$ where $\mathcal{W}, \mathbf{s}, \mathbf{f}$ are as defined above. Let $\Pi^*(\mathcal{J})$ denote an optimal solution of \mathcal{J} and let $\phi^*(\mathcal{J}) = \phi(\Pi^*(\mathcal{J}))$.

Our Contributions. The chapter contains the following three main results:

(A) *Hardness results.* In Section 6.2, we show that MRMP-RA is APX-hard under the weakly-monotone assumption. The NP-hardness of optimizing sum of distances (i.e., optimal MRMP) for the monotone and the general (non-monotone) case was shown in [GH22], but without revolving areas. Our main result here is the extension of the NP-hardness construction to prove that MRMP-RA, under the weakly-monotone assumption, is in fact APX-hard, which rules out a polynomial-time $(1 + \varepsilon)$ -approximation algorithm for it. To the best of our knowledge, this is the first APX-hardness result for any MRMP variant.

(B) *Approximation algorithm.* In Section 6.3, we present the first $O(1)$ -approximation algorithm that given an instance $\mathcal{J} = (\mathcal{W}, \mathbf{s}, \mathbf{f}, \mathcal{A})$ of MRMP-RA computes a feasible path ensemble Π from \mathbf{s} to \mathbf{f} such that Π is weakly-monotone and $\phi(\Pi) = O(1) \cdot \phi^*(\mathcal{J})$; note that $\Pi^*(\mathcal{J})$ need not be weakly-monotone, i.e., we approximate the general optimal path ensemble. In fact, we show that the robots can be moved in any order, so our algorithm can be extended to an online setting where the robots R_i , and their start/final positions, (s_i, f_i) are given in an online manner, or R_i 's may have to execute multiple tasks which are given in an online manner— the so-called life-long planning problem. Our algorithm ensures an $O(1)$ competitive ratio, i.e., the cost is $O(1)$ times the optimal cost of the offline problem.

The algorithm begins by computing a set of shortest paths Γ that avoid obstacles but ignore robot-robot collisions. Then, Γ is edited to avoid robot-robot collisions by moving non-active robots within their revolving areas. Our overall approach is the

²We use the term “weakly-monotone” because a plan is called monotone if the active interval of R_i is defined from when R_i leaves s_i for the first time until R_i reaches f_i for the last time, (rather than the leaving/reaching the revolving area A_{s_i}/A_{f_i}).

same as by Solomon and Halperin [SH18], but the editing of Γ differs significantly from [SH18], so that the cost of the paths does not increase by too much. We use a more conservative editing of Γ , which enables us to prove that the cost of the edited path ensemble is $O(1) \cdot \phi(\Gamma)$ (see Section 6.4), while the cost of the edited path in [SH18] is³ $O(\phi(\Gamma) + mn + m^2)$. Our main technical contributions are defining a more conservative retraction, proving that the motion plan remains feasible even under this conservative retraction, and bounding the total cost of the motion plan by using a combination of local and global arguments. Analyzing both the feasibility and the cost of the motion plan are nontrivial and require new ideas.

(C) Computing a good ordering. The result above shows that editing the paths increases the total cost of the motion plan only by a constant factor irrespective of the order in which we move the robots. However, the overhead in the overall cost due to editing (to avoid robot-robot collisions) can vary significantly depending on the ordering we chose. This raises the question whether we can find a “good” ordering that minimizes the overhead. The result in [SH18] implies that the problem of finding a good ordering that minimizes the amount of overhead is NP-hard.⁴ We present a polynomial time $O(\log n \log \log n)$ -approximation algorithm for finding a good ordering. This is achieved by reducing the problem to an instance of weighted feedback arc set in a directed graph, and applying an approximation algorithm for the latter problem [ENSS98]. This result is described in Section 6.5.

We emphasize that without additional, mitigating, assumptions, MRMP is intractable. Sampling-based planners assume that the full solution paths have some clearance around them—namely, each robot has some distance from the obstacles along its entire path, as well as from the other robots. Here, we assume certain clearance only at the start and goal positions; we do not make any assumption about the clearance along the paths. Indeed, we

³Notice that the roles of m and n here are reversed with respect to [SH18].

⁴The model in [SH18] for defining the overhead is different from ours, their construction can nevertheless be adapted to our setting.

assume non-negligible clearance, as we require that each robot at a start or goal position is encapsulated inside a disc of radius 2, which does not contain any other robot at its start or goal position. The choice of the number 2 here is not arbitrary. In a couple of related results for MRMP of unit discs [AdBHS15, BdBB⁺22] this is the critical value of clearance below which there does not always exist a solution to the problem.

6.2 Hardness of Optimal MRMP-RA

In this section we present our hardness results. Throughout this section all path ensembles are weakly-monotone, unless otherwise stated. With a slight abuse of notation we use \mathcal{C}^* to denote the cost of the optimal weakly-monotone path ensemble. Finding monotone path ensembles has been shown to be NP-hard in [GH21] using a similar grid-based construction without revolving areas.

NP-Hardness of weakly-monotone MRMP-RA Let $Q(x_1, \dots, x_n) = \bigwedge_{i=1}^m C_i$ be an instance of 3SAT with n variables and m clauses. Each clause C_i is a disjunction of three *literals*, which are variables or their negations. We construct a corresponding MRMP-RA instance $\mathcal{J} := \mathcal{J}(Q) = (\mathcal{W}, \mathbf{s}, \mathbf{f}, \mathcal{A})$ with $N = 3m + 1$ robots and choose a real value $d \geq 0$ such that $\mathcal{C}^*(\mathcal{J}) \leq d$ if and only if Q is satisfiable. Let $d(\mathcal{J}) := \sum_{i=1}^N d_i$, where d_i is the length of the optimal path of R_i from s_i to f_i in \mathcal{W} , ignoring other robots. In fact, our construction will choose d to be $d(\mathcal{J})$, that is, d is the lowest possible cost of a feasible path ensemble from \mathbf{s} to \mathbf{f} in \mathcal{W} . Our construction will ensure that the lowest cost is attained if and only if Q is satisfiable. \mathcal{J} is constructed so that a path ensemble with such a cost is possible if and only if (a feasible) monotone motion plan exists. An example of the construction is shown in Figure 6.2.

Overall description. The workspace \mathcal{W} consists of $m + n$ rectangular gadgets, one for each variable and each clause, referred to as *variable* and *clause* gadgets, respectively. All the gadgets have unit-width *passages* that are wider around revolving areas. For simplicity, the widened areas are shown as circular arcs, but they can easily be made polygonal. Each

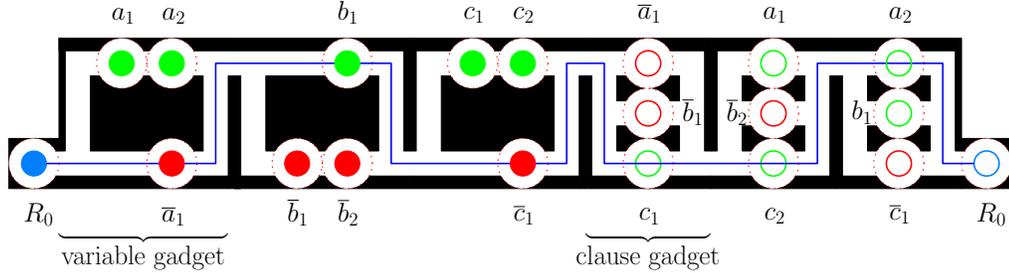


Figure 6.2: The MRMP-RA instance \mathcal{J} that corresponds to the formula $Q = (\bar{a} \vee \bar{b} \vee c) \wedge (a \vee \bar{b} \vee c) \wedge (a \vee b \vee \bar{c})$. The start and target positions are the filled and unfilled discs, respectively. Positive literal robots are green, negative literal robots are red. Obstacles appear in black. Start and target positions of literal robots are labeled with unique indices in order to distinguish between appearances of the same literal. The path π_0 is shown in blue for the assignment $a = T, b = F, c = T$ for which the corresponding path ensemble has robots moving in the following order: $\bar{c}_1, b_1, \bar{a}_1, r_0, a_2, a_1, \bar{b}_2, c_2, \bar{b}_1, c_1$.

gadget has an entrance on the left and an exit on the right. The vertical positions of entrances and exits alternate so that a gadget's entrance is connected to the exit of the gadget on its left.

There are $N = 3m + 1$ robots, each being a unit disc: one robot for each appearance of a literal in Q , which are collectively called *literal robots*, and one special *pivot* robot R_0 (shown in blue in Figure 6.2). The robot R_0 has to pass through all the gadgets from left to right, by which it is able to verify the satisfiability of Q , and the literal robots will constrain its motion in order to ensure that $\phi^*(\mathcal{J}) \leq d$.

Each variable (resp. clause) gadget contains two (resp. three) horizontal passages, which offer two (resp. three) shortest paths from its entrance to its exit. Each such path consists of vertical and horizontal line segments. The horizontal passages of the gadgets contain all the start and target positions of literal robots. All the revolving areas are centered at their respective start or target positions, and they do not overlap.

Gadgets. Each variable gadget initially contains robots representing literals of a single variable of Q . The top and bottom horizontal passages of the gadget contain robots representing only positive and negative literals, respectively. Each clause gadget has three horizontal passages, each containing a target position of one the literals in the correspond-

ing clause. The gadgets are placed within a horizontal strip from left to right such that variable gadgets are located to the left of clause gadgets. The order of gadgets of the same type is arbitrary, however it determines the order of the start positions, which is critical: the left to right order of start positions within each variable gadget is set to match the left to right order of the corresponding target positions. We refer to this order as the *intra-literal order property*. We say that a revolving area A is *congested* if it contains two robots at the same time. Intuitively, both optimal path ensembles and monotone path ensembles need to prevent revolving areas from becoming congested. We first establish that finding an optimal weakly-monotone path ensemble is equivalent to finding a monotone one, then show the equivalence between a satisfying assignment and a monotone path ensemble.

Lemma 37. *\mathcal{J} has a weakly monotone path ensemble with a cost of d if and only if \mathcal{J} has a monotone path ensemble.*

Proof. Let A be a revolving area in \mathcal{W} . We first note that without any loss of generality, in any path ensemble of \mathcal{J} a robot may either be contained in A at some point or never intersect A at all. That is, a robot will not partially penetrate a revolving area without ever fully entering it.

Let Π be a feasible path ensemble with $\phi(\Pi) = d$. We fix a robot R_i and examine the motion that occurs during its active interval τ_i . We claim that any motion of a robot $R_j, j \neq i$ during τ_i is *redundant*, i.e., if R_j does not move during τ_i then R_i can still perform the same motion. This suffices in order to conclude that Π can be made monotone. Observe that during the execution of Π no revolving area A can become congested, as otherwise the two robots that are simultaneously in A will have to take a path that is longer than the shortest path that ignores other robots. Therefore, whenever R_i is inside a revolving area A , it is the only robot in A , and any motion by other robots is redundant. Whenever R_i is not contained in any revolving area, all other robots must be contained in revolving areas, by definition. Hence, any motion by other robots at such point in time is also redundant. So overall, R_i may travel along its whole path without other robots moving.

For the other direction, in a monotone path ensemble it also holds that no revolving

area may become congested (as otherwise robots move simultaneously). Therefore, any revolving area that some robot R_i intersects during its motion must not contain other robots. For any gadget g that R_i needs to traverse, this allows R_i to take some shortest path through g . Therefore, R_i is able to take the shortest path that ignores other robots overall. Hence, a path ensemble with a cost of d exists. \square

Lemma 38. *Q has a satisfying assignment if and only if \mathcal{J} has a monotone path ensemble.*

Proof. Assume that Q has a satisfying assignment Λ . Let \mathcal{R}^+ (resp. \mathcal{R}^-) denote the set of robots corresponding to literals that evaluate to true (resp. false) according to Λ . That is, for each variable gadget, \mathcal{R}^+ contains robots that are all initially either in the top or the bottom passage, according to Λ . We show that the robots can move along optimal paths in the order $\mathcal{R}^-, R_0, \mathcal{R}^+$, which is made precise below.

Let π_0 be a shortest collision-free path from s_0 to f_0 that passes only through the start positions of \mathcal{R}^- and targets of \mathcal{R}^+ ; see Figure 6.2. The path π_0 exists because each clause gadget must contain a target of some robot in \mathcal{R}^+ , or else Λ does not satisfy Q .

In the path ensemble, each $R_i \in \mathcal{R}^-$ follows the subpath of π_0 from s_i (through which π_0 passes) up to the gadget containing f_i , from which R_i can reach its final position f_i using the shortest path. The order in which the robots in \mathcal{R}^- move is the right to left order of their start positions, which guarantees no collision with another robot located at its start position. Since the robots in \mathcal{R}^- move before \mathcal{R}^+ , the targets through which π_0 passes are unoccupied when the robots in \mathcal{R}^- move, guaranteeing no collisions at clause gadgets. Next, R_0 moves using π_0 , which passes through empty passages at this point. Finally, each $R_i \in \mathcal{R}^+$ joins π_0 at the vertical passage to its right, from which point it continues similarly to \mathcal{R}^- . The order of motion of the robots in \mathcal{R}^+ is the right to left order of their targets, which guarantees no collisions in the clause gadgets. Note that due to the intra-literal order property we also have no interferences among \mathcal{R}^+ within variable gadgets.

For the other direction, let us assume that there is a monotone path ensemble for \mathcal{J} . Let π_0 denote the path taken by R_0 . Without loss of generality, π_0 is weakly x -monotone.

Specifically, it passes through only one horizontal passage in each variable gadget. Therefore, we define an assignment Λ as follows: x is assigned to be true if and only if π_0 goes through the bottom passage of x 's variable gadget, which corresponds to negative literals. Let C be a clause of Q and let f_j be a target in C 's clause gadget that is unoccupied during R_0 's motion, which must exist. It is easy to verify that the literal corresponding to R_j is true according to Λ . Therefore, C is satisfied. \square

The construction can be carried out in polynomial time, therefore by combining Lemma 37 and Lemma 38 we obtain the following:

Theorem 39. *MRMP-RA for weakly-monotone path ensembles is NP-hard.*

Hardness of Approximation We now show that MRMP-RA is APX-hard, ruling out any polynomial time $(1 + \varepsilon)$ -approximation algorithm. We first go over some definitions. For an MRMP-RA instance \mathcal{J} , we use $\phi^*(\mathcal{J})$ to denote the cost of the optimal weakly-monotone path ensemble for \mathcal{J} . For a 3SAT formula Q , let $\text{SAT}(Q)$ denote the largest fraction of clauses in Q that can be simultaneously satisfied. We say that a revolving area A is *occupied* if it contains the robot whose start or target position lies in A .

To prove the hardness of approximation we present a gap-preserving reduction from MAX-3SAT(5), which is APX-hard [Vaz01]. The input to MAX-3SAT(5) is a 3SAT formula with 5 appearances for each variable and the goal is to find an assignment maximizing the number of satisfied clauses. Let Q be a MAX-3SAT(5) instance with n variables and m clauses and let $\mathcal{J} := \mathcal{J}(Q)$ be the MRMP-RA instance resulting from the NP-Hardness reduction described above, which we slightly modify as follows. Instead of the single pivot robot R_0 in \mathcal{J} , we now have m pivot robots. To this end, we modify the construction so that there is a horizontal passage that extends to the left of s_0 in I . The passage is lengthened to accommodate m start positions that lie on the same horizontal line, passing through s_0 in I . Similarly, another such passage is created to the right of f_0 to accommodate m target positions. The left to right order of the start positions of the pivot robots is set to match the left to right order of the corresponding target positions. Let \mathcal{J}' denote the resulting

MRMP-RA instance.

Lemma 40. *Let Q be a 3SAT formula such that for any assignment to Q there are at least k unsatisfied clauses in Q . Then $c^*(\mathcal{J}) > d(\mathcal{J}) + km$.*

Proof. Let us examine $\Pi^*(\mathcal{J})$, an optimal path ensemble for \mathcal{J} . We say that a robot R_i has a *bad event* during the execution of $\Pi^*(\mathcal{J})$ when R_i traverses an occupied revolving area. Note that each bad event results in R_i having a path longer than $1+d_i$, d_i being the length of R_i 's shortest possible path. We claim that each of the m pivot robots has k bad events, which suffices for proving the lemma.

Let us assume for a contradiction that one of the pivot robots, say R_i , has $q < k$ bad events. We will show how to obtain an assignment for Q where there are at most q unsatisfied clauses. Since $\Pi^*(\mathcal{J})$ is optimal, π_i , the path taken by R_i , is weakly x -monotone. We define an assignment Λ as follows (the same way as in the second direction of the proof of Theorem 38): x is assigned to be true if and only if π_i goes through the bottom passage of x 's variable gadget. In other words, Λ sets a literal to be true if and only if the corresponding literal-robot's starting position does not lie on π_i . Let us examine π_i right before it is R_i 's turn to move. Let \mathcal{R} denote the set of robots that are intersected by π_i and are located at variable gadgets at this point in time. We can assume without any loss of generality that \mathcal{R} is empty. If it is not, then let us examine the path ensemble Π where the robots in \mathcal{R} move to their targets before R_i 's turn. The number of bad events for R_i can only decrease in Π . This holds because by having some $R_j \in \mathcal{R}$ move before R_i we eliminate a bad event (for R_i) in R_j 's variable gadget and possibly introduce a bad event in R_j 's clause gadget.

Since there are q bad events for R_i , there are at most q clause gadgets where such an event occurs. Therefore, to get a contradiction it suffices to show that all other clause gadgets correspond to clauses that are satisfied by Λ . Let C be such a clause, i.e., in the corresponding clause gadget π_i passes through some empty revolving area A_{f_j} . Since π_i does not pass through any occupied revolving areas in the variable gadgets, the corresponding start position s_j must not lie on π_i . Therefore, r_j corresponds to a literal that is true by Λ , and so C is satisfied. \square

We now make $d(\mathcal{J}')$ explicit using an upper bound for an arbitrary d_i . First, we bound the length of each vertical segment in the corresponding path π_i by 10, which provides sufficient distance for our gadgets. Since each variable appears in Q five times, we bound the horizontal length of an variable gadget by $4 \cdot 2 + 3 = 11$ (i.e., there at most 4 revolving areas on a horizontal passage and some additional length). Therefore, the path length through any gadget is $O(1)$. Hence, we have $d_i = O(m)$ and the number of robots is also $O(m)$ (we have $m = 5n/3$). Therefore, we can set $d(\mathcal{J}') = cm^2$ for some sufficiently large constant c (we can easily lengthen paths in \mathcal{J}' if that is needed for the bound).

We can now combine the latter equality with Lemma 40 and the NP-Hardness reduction. Let us define $f(Q) := d(\mathcal{J}')$.

Theorem 41. *There is a polynomial time reduction that transforms an instance Q of $MAX-3SAT(5)$ with m clauses to an $MRMP-RA$ instance \mathcal{J}' such that $SAT(Q) = 1 \Rightarrow \phi^*(\mathcal{J}') = f(Q) \leq cm^2$ for some constant $c > 0$ and otherwise $SAT(Q) < \alpha \Rightarrow \phi^*(\mathcal{J}') > f(Q) + (1 - \alpha)m \cdot m = (1 + \frac{1-\alpha}{c}) f(Q)$, for all $0 < \alpha < 1$.*

6.3 Algorithm

Let $\mathcal{J} = (W, \mathbf{s}, \mathbf{f}, \mathcal{A})$ be an instance of $MRMP-RA$. Let n be the number of robots and m be the complexity of the environment W . We describe an $O(n(n+m) \log m)$ algorithm for computing a weakly-monotone path ensemble $\tilde{\Pi} := \tilde{\Pi}(\mathcal{J})$ for R_1, \dots, R_n such that $\phi(\tilde{\Pi}) = O(1) \cdot \phi^*(\mathcal{J})$. We remark that $\tilde{\Pi}$ is weakly-monotone but $\Pi^*(\mathcal{J})$ need not be, i.e. $\tilde{\Pi}$ is an $O(1)$ -approximation of any feasible motion plan. We parameterize the paths in $\tilde{\Pi}$ over the common interval $J = [0, n]$. We need a few definitions and concepts related to revolving areas. For any $z \in \mathbf{s} \cup \mathbf{f}$, let c_z denote the center of the revolving area A_z , and let C_z (resp. B_z) be the disc of radius 1 (resp. 3) centered at c_z , i.e., $C_z \subset A_z \subset B_z$. If $x \notin B_z$ then $D_x \cap A_z = \emptyset$. We refer to C_z and B_z as the *core* and *buffer*, respectively, of revolving area A_z . See Figure 6.3.

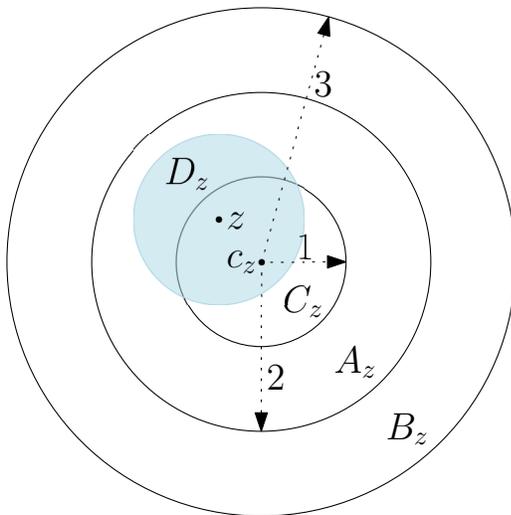


Figure 6.3: Revolving area A_z for some $z \in \mathbf{s} \cup \mathbf{f}$ with $D_z \subseteq A_z$, core C_z , and buffer B_z . (b) On the right, we show an instance of MRMP-RA. Each robot is shown as a filled disc in its starting revolving area, and its target revolving area is shown in the same color. Obstacles are dark gray.

Overview of the Algorithm. The algorithm consists of three stages. We note that Stage (I) and (II) are used in [SH18]. However, Stage (III) differs significantly from previous work in order to ensure the total cost of paths is within an $O(1)$ factor of that of the optimal motion plan. We describe all stages for completeness.

I. We compute the free space \mathcal{F} (with respect to a single robot) using the algorithm of Ó’Dúnlaing and Yap [Yap87, OY85]. If s_i and f_i , for some $i \in [n] := \{1, 2, \dots, n\}$, do not lie in the same connected component of \mathcal{F} , then a feasible path does not exist for R_i from s_i to f_i . Therefore, we stop and return that no feasible motion plan exists from \mathbf{s} to \mathbf{f} . Next, for each i , we compute a shortest path γ_i from s_i to f_i , ignoring other robots using the algorithm of Chen and Wang [CW15]. Let $\Gamma = \{\gamma_1, \dots, \gamma_n\}$ be the path ensemble computed by the algorithm.

Although Γ does not intersect \mathcal{O} , it may not be feasible since two robots may collide during the motion. The next two steps deform Γ to convert it into a feasible motion plan. We take an arbitrary permutation σ of $[n]$. Without loss of generality assume $\sigma = \langle 1, 2, \dots, n \rangle$. We say that R_i is *active* during the subinterval $[i-1, i]$ of $J := [0, n]$,

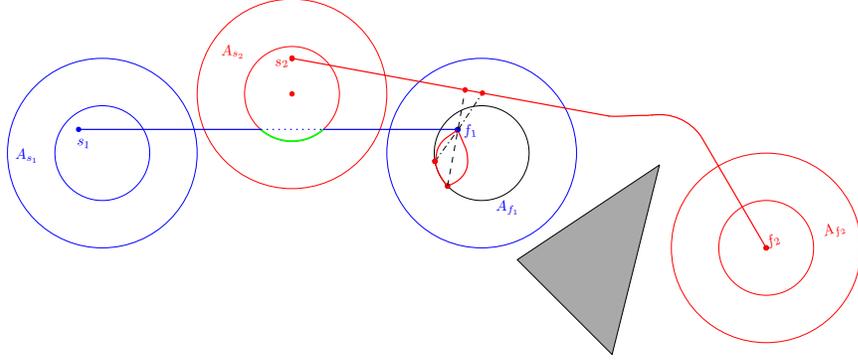


Figure 6.4: Path γ_1 is shown in blue from s_1 to f_1 . Assume R_1 is active before R_2 . In $\bar{\gamma}_1$, the dotted portion of the path is replaced with the green arc along ∂C_{s_2} . Path π_2 is shown in red from s_2 to f_2 . R_1 must follow the red retraction during the movement of R_2 in B_{f_2} .

during which it moves from s_i to f_i . During $[0, i - 1]$ (resp. $[i, n]$) R_i only moves within the revolving area A_{s_i} (resp. A_{f_i}).

II. For each i , we first modify γ_i , as described below in Section 6.3.1, so that it does not intersect the interior of the core C_j of any revolving area A_j that is occupied by a robot R_j , for $j \neq i$; see Figure 6.4. Let $\bar{\gamma}_i$ be the deformed path. Abusing the notation a little, let $\bar{\gamma}_i : [i - 1, i] \rightarrow \mathcal{F}$ denote a uniform parameterization of the path $\bar{\gamma}_i$, i.e. R_i moves with a fixed speed during $[i - 1, i]$ from s_i to f_i along $\bar{\gamma}_i$. We extend $\bar{\gamma}_i$ to the interval $[0, n]$ by setting $\bar{\gamma}_i(t) = s_i$ for $t \in [0, i - 1]$ and $\bar{\gamma}_i(t) = f_i$ for $t \in [i, n]$. Set $\bar{\Gamma} = \{\bar{\gamma}_1, \dots, \bar{\gamma}_n\}$.

III. Next, for each distinct pair $i, j \in [n]$, we construct a *retraction map* $\rho_{ij} : \mathcal{F} \rightarrow \mathcal{F}$ that specifies the position of R_j for a given position of R_i during the interval $[i - 1, i]$ when R_i is active so that R_i and R_j do not collide as R_i moves along $\bar{\gamma}_i$. The retraction map ensures that R_j stays within the revolving area A_{s_j} (resp. A_{f_j}) for $j < i$ (resp. $j > i$), and it does not collide with any R_k for $k \neq i, j$, as well. See Figure 6.4. Using this retraction map, we construct the path $\pi_j : J \rightarrow \mathcal{F}$ as follows:

$$\pi_j(t) = \begin{cases} \rho_{ij}(\bar{\gamma}_i(t)) & \text{for } t \in [i - 1, i] \text{ and } i \neq j, \\ \bar{\gamma}_j(t) & \text{for } t \in [j - 1, j]. \end{cases}$$

We prove below that each π_j is a continuous path. In Section 6.4, we prove that $\Pi = \{\pi_1, \dots, \pi_n\}$ is a feasible path ensemble with $\varphi(\Pi) = O(1) \cdot \varphi^*(\mathcal{J})$.

6.3.1 Modifying initial path

Fix an $i \in [n]$. For $j < i$, let $z_j = f_j$ and for $j > i$, let $z_j = s_j$. Set $Z = \{z_j : 1 \leq j \neq i \leq n\}$. This step modifies γ_i to ensure that the path of R_i does not enter the core C_z of any $z \in Z$.

Fix a $z \in Z$. If $\gamma_i \cap C_z = \emptyset$, then R_j does not affect γ_i . If $\gamma_i \cap C_z \neq \emptyset$, then we modify γ_i as follows: let p_z, q_z be the first and last intersection points of γ_i and C_z along γ_i , respectively. Let Q_z be the shorter arc of ∂C_z , the boundary of the core C_z , between p_z and q_z . We replace $\gamma_i[p_z, q_z]$ with Q_z . We repeat this step for all $z \in Z$. Let $\bar{\gamma}_i$ be the resulting path from s_i to f_i ; $\bar{\gamma}_i$ does not intersect $\text{int}(C_z)$ for any $z \in Z$. Note that C_{z_j} 's are pairwise disjoint, and that γ_i is a shortest path from s_i to f_i in \mathcal{F} , therefore $\gamma_i[p_z, q_z]$ and $\gamma_i[p_{z'}, q_{z'}]$, for any pair $z, z' \in Z$, are disjoint. We can thus process Z in an arbitrary order and the resulting path does not depend on the ordering. Furthermore $C_z \subseteq \mathcal{F}$ (since $A_z \subseteq \mathcal{W}$), so $\bar{\gamma}_i \subset \mathcal{F}$ for all i .

6.3.2 Retracting a robot

We now describe the retraction motion of R_j when R_i is active, so that they do not collide. Note that for all $t \in [i-1, i]$, $\bar{\gamma}_j(t) = z_j$, i.e., before applying the retraction R_j is at z_j when R_i is active. We define the *retraction function* $\rho_{ij} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ that specifies the motion of R_j within A_j during $[i-1, i]$. Since i is fixed, for simplicity we use ρ_j to denote ρ_{ij} , and we use C_j (resp. A_j, B_j) for disc C_{z_j} (resp. A_{z_j}, B_{z_j}). If the center of R_i is at distance at least 2 from z_j , then R_i does not intersect D_j , so there is no need to move R_j from z_j . Therefore we set $\rho_j(p) = z_j$ for all $p \in \pi_i$ such that $\|p - z_j\| \geq 2$. On the other hand, $\bar{\gamma}_i$ does not intersect the interior of C_j so $\rho_j(p)$ is undefined for $p \in \text{int}(C_j)$. We thus focus on the case when $\|p - z_j\| \leq 2$, in which case p lies in the buffer disc B_j , and $p \notin \text{int}(C_j)$, i.e., $p \in B_j \setminus \text{int}(C_j)$.

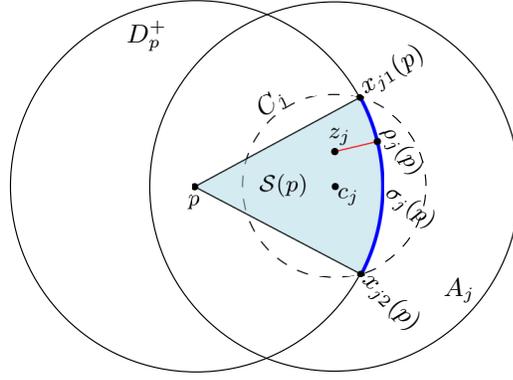


Figure 6.5: Retraction Map: A sector type retraction (when z_j lies in $S(p)$).

Let $D^+(p)$ be the disc of radius 2 centered at p . Note that for a point $q \in \mathbb{R}^2$, $\text{int}(D_p) \cap \text{int}(D_q) \neq \emptyset$ if and only if $q \in D^+(p)$. Intuitively, we move the center of R_j from z_j (within C_j) as little as possible so that R_j does not collide with $R_i(p)$. Formally, we define ρ_j as: $\rho_j(p) = \text{argmin}_{q \in C_j \setminus D_p^+} \|q - z_j\|$ if $p \notin \text{int}(C_j)$, and undefined otherwise.

In the remainder of the discussion, we assume $\|z_j - p\| \leq 2$ and $p \notin C_j$, so $p \in B_j \setminus C_j$. Therefore, $\rho_j(p)$ exists and additionally $\rho_j(p)$ is unique. We now discuss the two possible types of retraction. Refer to Figure 6.5 and 6.6 throughout this paragraph. Note that ∂C_j and ∂D_p^+ intersect at exactly two points since $p \in B_j \setminus C_j$, say, $x_{j1}(p), x_{j2}(p)$. Let $\sigma_j(p)$ be the smaller of the two arcs of ∂D_p^+ induced by $x_{j1}(p)$ and $x_{j2}(p)$, and let $S(p) = \text{conv}(\sigma_j(p) \cup \{p\}) \subseteq D_p^+$ be the *sector* of $D^+(p)$ induced by $x_{j1}(p), x_{j2}(p)$. Observe that the retraction point $\rho_j(p)$ lies on $\sigma_j(p)$.

If $z_j \in S(p)$, then $\rho_j(p)$ is the intersection point of the ray $\overrightarrow{pz_j}$ with ∂D_p^+ , as the closest point in $C_j \setminus D_p^+$ from z_j is on the straight line from z_j to D_p^+ . Since z_j lies inside $S(p)$, $\rho_j(p) \in \partial S(p)$. See Figure 6.5.

If $z_j \notin S(p)$, the retraction point is $\text{argmin}_{q \in \{x_{j1}(p), x_{j2}(p)\}} \|q - z_j\|$, i.e., the closest point to z_j in $C_j \setminus D_p^+$ is an endpoint of $\sigma_j(p)$. See Figure 6.6. Note that our retraction ensures that R_j will be centered back at z_j after robot R_i moves away.

In the remainder of the chapter, if $\rho_j(p) \in \{x_{j1}(p), x_{j2}(p)\}$ we say that the retraction is of *intersection* type, otherwise we say that the retraction is of *sector* type. Since $\rho_j(p) \in$

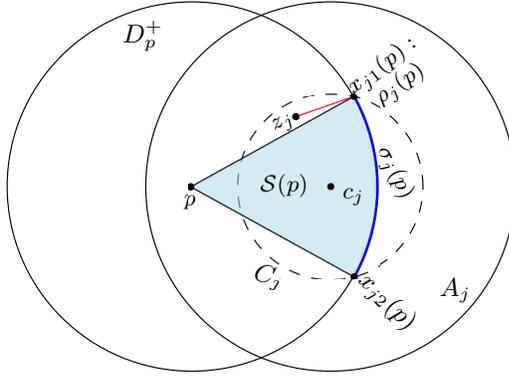


Figure 6.6: Retraction Map: an intersection type retraction, z_j lies outside $S(p)$, the retraction point is $x_{j1}(p)$.

C_j for all $p \notin C_j$ and none of the $\bar{\gamma}_i$'s enter C_j , the retraction path π_j of R_j lies in \mathcal{F} . We conclude this section with the following lemma, which follows from the fact that ρ_j is a continuous function, $\rho_j(p) = z_j$ for all p such that $\|p - z_j\| \geq 2$, and $\|z_j - s_i\|, \|z_j - f_i\| \geq 2$.

Lemma 42. *For any $1 \leq i \leq n$, π_i is a continuous path from s_i to f_i .*

6.4 Correctness and Analysis of the Algorithm

We first prove that Π is feasible (Section 6.4.1), then we bound $\phi(\Pi)$ (Section 6.4.2), and finally analyze the running time in Section 6.4.3. We begin by summarizing a few relevant properties of revolving areas (see Figure 6.7), which are straightforward to prove.

Lemma 43. *Let $x, y \in \mathbf{s} \cup \mathbf{f}$ such that $x \neq y$:*

1. $x \in C_x$, that is, each start or final position lies inside the core of A_x ;
2. $\|c_x - c_y\| \geq 2$, i.e., $\text{int } C_x \cap \text{int } C_y = \emptyset$;
3. for any $p \in C_x$, $\|p - y\| \geq 2$, i.e., $\text{int } D_p \cap \text{int } D_y = \emptyset$;
4. $\|x - c_y\| \geq 3$, i.e., each start/final position lies outside the buffer of any other start/final position.

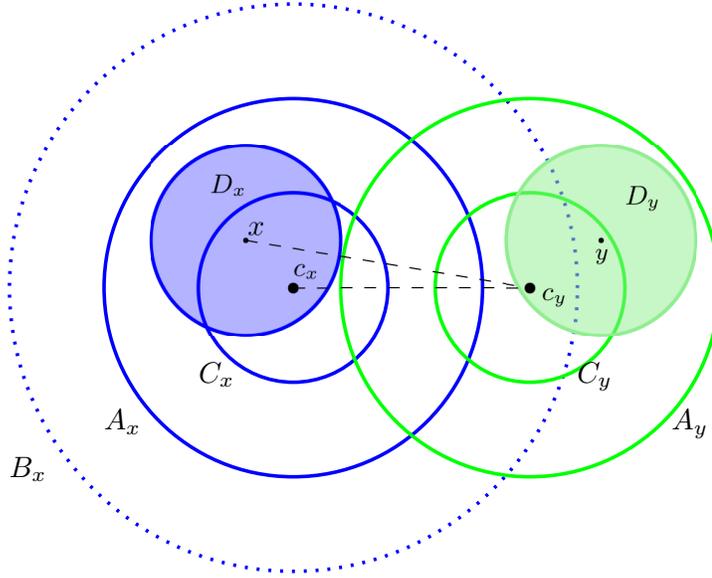


Figure 6.7: Illustration of Lemma 43. D_x and D_y are two robots located in their respective revolving areas A_x, A_y . The distance between c_x and c_y is at least 2; y lies outside the buffer of x , i.e., $y \notin B_x$.

6.4.1 Feasibility

In this section, we show that path ensemble II is feasible. Recall that stage (I) of the algorithm reports that there is no feasible solution if any $s_i \in \mathbf{s}$ and $f_i \in \mathbf{f}$ do not lie in the same connected component. So assume that Stage I computes a feasible path γ_i for each R_i . Stages II and III modify these paths so that they remain in \mathcal{F} . Hence, we only need to show that no two robots collide with each other during the motion, i.e., for any $1 \leq i \neq j \leq n$ and for any $t \in J$, $\text{int } D_{\pi_i(t)} \cap \text{int } D_{\pi_j(t)} = \emptyset$. We fix some $i \in [n]$ and the corresponding active interval $T_i := [i - 1, i] \subseteq J$ and prove the feasibility of II during this interval. Note that R_i is the only active robot in T_i and other robots stay in their revolving areas. By the definition of retraction, for any $t \in T_i$, and for any $j \neq i$, $\|\pi_i(t) - \rho_{ij}(\pi_i(t))\| \geq 2$, so R_i does not collide with R_j during interval T_i .

Lemma 44. For any $j \neq k$ and $z_j, z_k \in \mathbf{s} \cup \mathbf{f}$, the minimum distance between the line segments $c_j z_j$ and $c_k z_k$ is at least 2, i.e., $\min_{\substack{y_j \in c_j z_j, \\ y_k \in c_k z_k}} \|y_j - y_k\| \geq 2$.

Proof. Let y_j, y_k be the closest pair of points on the segments $c_j z_j$ and $c_k z_k$. Note that

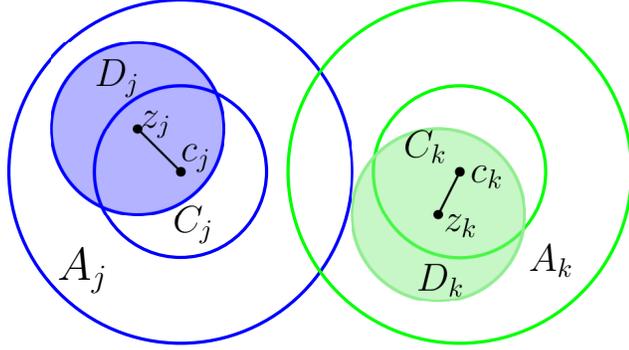


Figure 6.8: Illustration of Lemma 44.

$z_j c_j$ and $z_k c_k$ are disjoint since $z_j c_j \in C_j$ and $z_k c_k \in C_k$ and these cores do not intersect (cf Lemma 43). This implies that either y_j or y_k must be an endpoint of the respective segment.

Assume without loss of generality that y_j is an endpoint of $z_j c_j$. By Lemma 43, $\|c_j - z_k\|, \|c_k - z_j\| \geq 3$. Let y'_k be the endpoint of $c_k z_k$ at distance 3 from y_j ($y'_k = z_k$ if $y_j = c_j$ and $y'_k = c_k$ otherwise). Since $z_k \in C_k$, $\|y_k - y'_k\| \leq 1$, Then $\|y_j - y_k\| \geq \|y_j - y'_k\| - \|y'_k - y_k\| \geq 3 - 1 = 2$. \square

Lemma 45. *For any $j \neq k$, R_j and R_k do not collide during the interval T .*

Proof. In view of the above discussion, we assume $j, k \neq i$. The claim is equivalent to showing that $\|\rho_j(\pi_i(t)) - \rho_k(\pi_i(t))\| \geq 2$ for every $t \in T$. Let $p = \pi_i(t)$. There are two cases:

Case 1: $\rho_j(p) = z_j$ or $\rho_k(p) = z_k$. Without loss of generality, assume that $\rho_j(p) = z_j$. By construction, $\rho_k(p) \in C_k$, therefore by Lemma 43(iii), $\|\rho_j(p) - \rho_k(p)\| = \|z_j - \rho_k(p)\| \geq 2$.

Case 2: $\rho_j(p) \neq z_j$ and $\rho_k(p) \neq z_k$. Recall D_p^+ is the disc of radius 2 centered at p , and $x_{j,1}, x_{j,2}$ are the intersection points of the core of A_j and ∂D_p^+ .

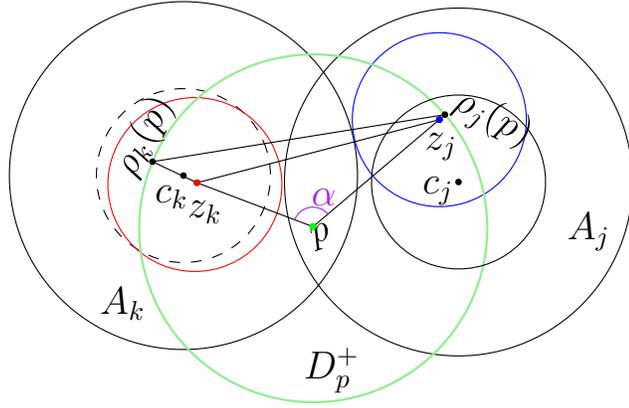


Figure 6.9: Illustration of Lemma 45, The angle $\alpha \geq \pi/3$.

We consider the triangle formed by the retraction points $\rho_k(p)$, $\rho_j(p)$ and p . We show that $\angle \rho_k(p)p\rho_j(p) \geq \pi/3$. We will first define a point $f_j(p)$ based on the current retraction type of R_j .

If the retraction of R_j is type sector, then z_j lies within the sector $S(p)$, let $f_j(p) = z_j$. Otherwise, the retraction is of type intersection and without loss of generality we assume $\rho_j(p)$ is $x_{j,1}(p)$. In this case, consider the segments $px_{j,1}(p)$ and $z_j c_j$. These two segments must intersect, as $c_j \in S(p)$ and $z_j \notin S(p)$. We let $f_j(p)$ be the intersection point of segments. Note that in either case, $f_j(p)$ lies on segment $p\rho_j(p)$. We analogously define $f_k(p)$. See Figure 6.9 for an example where $f_j(p) = z_j$ and $f_k(p) = z_k$.

By definition, $f_j(p) \in z_j c_j$ and $f_k(p) \in z_k c_k$ and Lemma 44 implies that $\|f_j(p) - f_k(p)\| \geq 2$. Additionally, $f_j(p) \in D_p^+$, so $\|p - f_j(p)\| \leq 2$ (similarly $\|p - f_k(p)\| \leq 2$).

Let α be the angle $\angle f_k(p)p f_j(p)$. Since $\|p - f_j(p)\|, \|p - f_k(p)\| \leq 2$, and $\|f_j(p) - f_k(p)\| \geq 2$, $\alpha \geq \pi/3$. Now consider the triangle formed by the retraction points and p . By construction, $\angle f_k(p)p f_j(p) = \angle \rho_k(p)p\rho_j(p)$. The distance between p and each retraction point is 2: $|p\rho_j(p)| = |p\rho_k(p)| = 2$. This implies the other two angles in the triangle are equal ($\angle p\rho_k(p)\rho_j(p) = \angle p\rho_j(p)\rho_k(p)$). Since $\angle \rho_k(p)p\rho_j(p) = \alpha \geq \pi/3$, $\rho_j(p)\rho_k(p)$ is the longest edge of the triangle $\triangle p\rho_j(p)\rho_k(p)$. The other two sides have length 2, so $\|\rho_j(p) - \rho_k(p)\| \geq 2$,

as desired. □

Since Lemma 45 holds for any interval T_i for $i \in [n]$, we obtain the following corollary.

Corollary 46. *The path ensemble Π returned by the algorithm is feasible.*

6.4.2 Cost of path ensemble

In this section, we analyze the cost of the path ensemble Π the algorithm returns. Stage (I) of the algorithm computes Γ , the shortest paths of all robots in \mathcal{F} while ignoring other robots. Clearly, we have $\phi(\Gamma) \leq \phi^*(\mathcal{J})$. We will show that $\phi(\Pi) = O(\phi(\Gamma))$.

Cost of $\bar{\Gamma}$. Stage (II) of the algorithm deforms Γ to $\bar{\Gamma}$. Path $\gamma_i \neq \bar{\gamma}_i$ only if $\gamma_i \cap \text{int } C_j \neq \emptyset$ for some $j \neq i$, otherwise $\ell(\gamma_i) = \ell(\bar{\gamma}_i)$. Suppose $\gamma_i \cap C_j \neq \emptyset$ for some $j \neq i$. Then in $\bar{\gamma}_i$, $\gamma_i \cap C_j$ is replaced with the shorter arc σ of ∂C_z , determined by the first and last endpoints, say p and q , of $\gamma_i \cap \partial C_j$. Therefore, $\ell(\sigma) \leq 2 \sin^{-1} \left(\frac{\|p-q\|}{2} \right) \leq 2\ell(C_j \cap \gamma_i)$. Hence, $\ell(\bar{\gamma}_i) \leq 2\ell(\gamma_i)$ and we obtain the following bound on $\phi(\bar{\Gamma})$.

Lemma 47. $\phi(\bar{\Gamma}) \leq 2\phi(\Gamma)$.

We now focus on bounding the length of retraction paths of non-active robots. Let $\pi_{ji} = \pi_j[i-1, i]$, and let $\Delta_{ij} = \{t \in [i-1, i] : \|\pi_i(t) - z_j\| \leq 2\}$, i.e., π_{ji} is the retraction of R_j due to the motion of R_i and $\pi_i[\Delta_{ij}]$ is the part of π_i that causes the retraction motion of R_j . Refer to Figure 6.4. We show that $\ell(\pi_{ji}) = O(\ell(\pi_i[\Delta_{ij}]))$ (cf Corollary 52) and charge π_{ji} to $\pi_i[\Delta_{ij}]$. We bound $\ell(\pi_{ji})$ by splitting into two scenarios. Roughly speaking, if π_i does not penetrate the buffer B_j too deeply, we use a Lipschitz condition on the retraction map to show $\ell(\pi_{ji}) = O(\ell(\pi_i[\Delta_{ij}]))$. More concretely, for $z \in \mathbf{s} \cup \mathbf{f}$, let W_z be the disk of radius $3/2$ centered at c_j . We prove a Lipschitz condition when the active robot lies outside W_j (cf Corollary 50). On the other hand, if π_i travels into W_j then the Lipschitz condition may not hold, but we argue that $\ell(\pi_i[\Delta_{ij}]) = \Omega(1)$ and that $\ell(\pi_{ji}) = O(1)$ (cf Lemma 51). Finally, using a packing argument, we show that each “point” of π_i is only charged $O(1)$ times, and thus $\phi(\Pi) = O(\phi(\bar{\Gamma})) = O(\phi(\Gamma))$.

Retraction of R_j outside W_j . As in Section 6.4.1, we fix an interval $[i-1, i]$ for some $i \in [n] \setminus \{j\}$. Let $\Delta_j^o = \{t \in [i-1, i] : \|\pi_i(t) - z_j\| \leq 2 \text{ and } \pi_i(t) \notin W_j\}$. That is, Δ_j^o is the interval(s) of time in which the path of robot R_i forces the retraction of robot R_j while the center of R_i lies outside W_i . Let Φ_{ij} be the restriction of path π_i of robot R_i during the interval Δ_j^o , i.e. $\Phi_{ij}(t) = \pi_i(t)$ for $t \in \Delta_j^o$. Let $\Psi_{ji} : \Delta_j^o \subset \Delta_{ij} \rightarrow C_j$ be the retraction of R_j during Δ_j^o , i.e., $\Psi_{ji}(t) = \rho_{ij}(\pi_i(t))$ for $t \in \Delta_j^o$. We show that $\ell(\Psi_{ji}) = O(\ell(\Phi_{ij}))$ by proving a Lipschitz condition on $\ell(\Psi_{ji})$.

We will divide Φ_{ij} into subpaths, referred to as pathlets, so that there is only one type of retraction point associated with the subpath. We call a time instance $t \in \Delta_j^o$ an *event* if t is either an endpoint of a connected component of Δ_j^o (i.e., $\|\pi_i(t) - c_j\| = 3/2$ or $\|\pi_i(t) - z_j\| = 2$) or $z_j \in \partial S_j(\pi_i(t))$, (i.e., the type of retraction point $\rho_j(\pi_i(t))$ changes at time t). Let $t_0 < t_1 < \dots < t_k$ be the event points. We divide Φ_{ij} and Ψ_{ji} into *pathlets* at these events, i.e., $\Phi_{ij} = \varphi_1 \circ \varphi_2 \circ \dots \circ \varphi_g$ and $\Psi_{ji} = \psi_1 \circ \psi_2 \circ \dots \circ \psi_g$ where $\varphi_k = \pi_i[t_{k-1}, t_k]$ and $\psi_k = \rho_j(\varphi_k) = \pi_j[t_{k-1}, t_k]$. We prove the Lipschitz condition for each pathlet. All points on $\rho_j(\varphi_k)$ have the same type of retraction by construction of Φ_{ji} . We call φ_k a *sector-type* (*intersection-type*) pathlet if all points have sector (resp. intersection) type retraction.

Lemma 48. *For a sector-type pathlet φ_k of Φ_{ji} , $\ell(\rho_j(\varphi_k)) = O(\ell(\varphi_k))$.*

Proof. For each $p \in \varphi_k$, $\rho_j(p)$ is type sector, i.e. $\rho_j(p)$ lies on the ray $\overrightarrow{pz_j}$ at distance 2 from p . In this case, the retraction map $\psi_k = \rho_j(\varphi_k)$ traces a portion of a Conchoid [SS83].

We parameterize points on $\varphi := \varphi_k$ and $\psi := \psi_k$ using polar coordinates, with z_j as the origin. Let $\varphi(\theta) = (r(\theta), \theta)$ be a point on φ , where θ is the orientation of the point with respect to the x -axis (with z_j as the origin). Then, $\psi(\theta) = \rho_j(\varphi(\theta)) = (2 - r(\theta), \theta)$. See Figure 6.10. Note that $\|\varphi'(\theta)\|^2 = r^2(\theta) + (r'(\theta))^2$ and $\|\psi'(\theta)\|^2 = (2 - r(\theta))^2 + (r'(\theta))^2$. Since φ lies outside W_j and $z_j \in C_j$, we have $r(\theta) \in [1/2, 2]$. Therefore, $2 - r(\theta) \leq 3r(\theta)$ and $\|\psi'(\theta)\| \leq 3\|\varphi'(\theta)\|$. Hence,

$$\ell(\psi) = \int \|\psi'(\theta)\| d\theta \leq 3 \int \|\varphi'(\theta)\| d\theta = 3\ell(\varphi). \quad \square$$

Lemma 49. *For an intersection-type pathlet φ_k of Φ_{ji} , $\ell(\rho_j(\varphi_k)) = O(\ell(\varphi_k))$.*

Proof. Again, we prove the lemma by showing that a Lipschitz condition holds. Let $\varphi := \varphi_k$. We parameterize both φ and $\rho_j(\varphi)$ in polar coordinates, but with c_j as the origin. Let $I = [a, b]$ be the interval over which φ is defined. Let $\varphi(t) = (r(t), \theta(t))$ for $t \in I$. We assume that φ is sufficiently small (otherwise we divide it into smaller pathlets and argue for each pathlet) so that φ both r - and θ -monotone.

Set $\Delta\varphi_r = |r(b) - r(a)|$ and $\Delta\varphi_\theta = |\theta(b) - \theta(a)|$. Since φ lies outside W_j , $r(t) \geq 3/2$ for all $t \in [a, b]$. W.l.o.g., assume both $r(t)$ and $\theta(t)$ are monotonically non-decreasing. We obtain:

$$\ell(\varphi) = \int_I \sqrt{r'(t)^2 + r(t)\theta'(t)^2} \geq \frac{1}{\sqrt{2}} \int_I \left(r'(t) + \frac{3}{2}\theta'(t) \right) dt \geq \frac{1}{\sqrt{2}}(\Delta\varphi_r + \Delta\varphi_\theta).$$

The retraction path $\psi(t)$ varies monotonically on the unit circle ∂C_z . Thus, we parameterize ψ by its direction on ∂C_z , and $\ell(\psi) = |\int_I \psi'(t) dt| = |\psi(b) - \psi(a)|$. To bound $\ell(\psi)$, consider the following path from $\varphi(a)$ to $\varphi(b)$, see Figure 6.11. Let $\bar{\varphi}_1$ be the arc from $\varphi(a)$ to point $\xi = (r(a), \theta(b))$ along the circle of radius $r(a)$ centered at c_z . Let $\bar{\varphi}_2$ be the segment ξ to $\varphi(b)$, this is a radial segment on line ξc_z . Then, $\ell(\psi) \leq \ell(\rho_j(\bar{\varphi}_1)) + \ell(\rho_j(\bar{\varphi}_2))$. Since the radius along $\bar{\varphi}_1$ does not change, $\ell(\rho_j(\bar{\varphi}_1)) = |\theta(b) - \theta(a)| = \Delta\varphi_\theta$.

For a point $p = (r, \theta)$, the orientation of $\rho_j(p)$ is $\theta + \cos^{-1}\left(\frac{3-r^2}{2r}\right)$ (by the law of cosines, considering triangle $\Delta\rho_j(p)c_z p$). Since θ does not change along $\bar{\varphi}_2$ and $r(a), r(b) \in [3/2, 3]$, we obtain $\ell(\rho_j(\bar{\varphi}_2)) = O(\Delta\varphi_r)$.

Putting everything together, $\ell(\rho_j(\psi)) = \ell(\psi) = O(\Delta\varphi_r + \Delta\varphi_\theta) = O(\ell(\varphi))$. □

Applying Lemmas 48 and 49 to all pathlets of Φ_{ij} , we obtain the following:

Corollary 50. *Let $1 \leq i \neq j \leq n$. Let Φ_{ij} be the portion of π_i during the interval $t \in [i-1, i]$ such that $\|\pi_i(t) - z_j\| \leq 2$ and $\|\pi_i(t) - c_j\| \geq 3/2$, and let Ψ_{ji} be the retraction of R_j corresponding to Φ_{ij} . Then $\ell(\Psi_{ji}) = O(\ell(\Phi_{ij}))$.*

Retraction path inside W_j . Recall that π_i does not intersect (int C_j), but possibly travels along ∂C_j . For a point $p \in \pi_i$, if $p \in \partial C_j$, then $\rho_j(p)$ is the point on ∂C_j diametrically opposite p . Thus, $\ell(\pi_i \cap C_j) = \ell(\rho_j(\pi_i \cap C_j))$. In the following, we consider only $\pi_i \setminus \partial C_j$.

Lemma 51. *For a pathlet φ (i.e., a connected subpath) of path π_i such that $\varphi \subset W_j \setminus C_j$ for some $j \neq i$, $\ell(\rho_j(\varphi)) = O(\ell(\pi_i \cap A_j))$.*

Proof. Since $\varphi \subset W_j$, $\ell(\pi_i \cap A_j) = \Omega(1)$, therefore we only need to argue that $\ell(\rho_j(\varphi))$ is constant. We will bound the length of both types of retraction maps (intersection and sector) separately for φ , and use the sum as an upper bound on the length of the actual retraction map.

Sector retraction. We consider the sector type retraction map. Let z_j be the origin and consider polar coordinates. Let $\rho_j^s(p)$ be the sector type retraction point with respect to p . Since φ is a subpath of a shortest path in \mathcal{F} , we can divide $\pi_i \cap W_j$ into at most two pathlets such that each piece is r, θ -monotone. Abusing notation, let φ be one of these pieces with endpoints (r_0, θ_0) and (r_1, θ_1) .

We write the retraction point parameterized by θ as $(\rho(\theta), \theta)$. Using the fact that $\rho(\theta) \leq 2$ for all θ , the arc length of the retraction map is

$$\begin{aligned} \ell(\rho_j^s(\varphi)) &= \int_{\theta_0}^{\theta_1} \sqrt{\rho(\theta)^2 + \left(\frac{d\rho}{d\theta}\right)^2} d\theta \leq \int_{\theta_0}^{\theta_1} \rho(\theta) d\theta + \int_{\theta_0}^{\theta_1} \frac{d\rho(\theta)}{d\theta} d\theta \\ &\leq \rho(\theta_1 - \theta_0) + (\rho(\theta_1) - \rho(\theta_0)) \leq 2(\theta_1 - \theta_0) + 2. \end{aligned}$$

Therefore, $\ell(\rho_j^s(\varphi)) = O(1)$, for each φ .

Intersection retraction. We consider the retraction map defined by an intersection point of ∂D_p^+ and ∂C_j . We now let c_j be the origin and consider polar coordinates. Let $\rho_j^i(p)$ be the intersection type retraction point closest to z_j with respect to p . Again, we divide $\pi_i \cap W_j$ into at most two pathlets such that each of them is r, θ -monotone (one pathlet is the portion of π_i coming closer to the core C_j , and the other moves away from C_j). Let φ be one of the pathlets with endpoints (r_0, θ_0) and (r_1, θ_1) . The retraction point lies on the unit circle ∂C_j , and as θ changes monotonically from θ_0 to θ_1 , the retraction point $\rho_j^i(\theta)$ moves monotonically on ∂C_j . Therefore, $\ell(\rho_j^i(\varphi)) = O(1)$.

Finally, $\ell(\rho_j(\varphi)) \leq \ell(\rho_j^s(\varphi)) + \ell(\rho_j^i(\varphi)) = O(1)$. □

Applying Lemma 51 to each of (at most two) connected components of $(\pi_i \cap W_j) \setminus C_j$ and combining with Corollary 50, we obtain the following:

Corollary 52. For $1 \leq i \neq j \leq n$, let Δ_{ij} be defined as $\Delta_{ij} = \{t \in [i-1, i] : \|\pi_i - z_j\| \leq 2\}$ and let $\pi_{ji} = \pi_j[i-1, i]$. Then $\ell(\pi_{ji}) = O(\ell(\pi_i[\Delta_{ij}]))$.

Cost of Path Ensemble. We are now ready to bound the cost of the path ensemble Π returned by the algorithm.

Lemma 53. For an instance \mathcal{J} of optimal MRMP with revolving areas, let $\Pi(\mathcal{J})$ be the path ensemble returned by the algorithm. Then $\phi(\Pi(\mathcal{J})) = O(1) \cdot \phi^*(\mathcal{J})$.

Proof. Set $\Pi = \Pi(\mathcal{J})$. We already argued that $\phi(\bar{\Gamma}) = O(\phi^*(\mathcal{J}))$, where $\bar{\Gamma}$ is the path ensemble computed in stage II of the algorithm. We thus need to prove $\phi(\Pi) = O(\phi(\bar{\Gamma}))$. For a pair $1 \leq i, j \leq n$, let $\pi_{ij} = \pi_i[j-1, j]$. By construction, $\ell(\pi_{ii}) = \ell(\bar{\gamma}_i)$. For a fixed i ,

$$\ell(\pi_i) = \sum_{j=1}^n \ell(\pi_{ij}) = \ell(\bar{\gamma}_i) + \sum_{j \neq i} \ell(\pi_{ij}) = \ell(\bar{\gamma}_i) + \sum_{j \neq i} O(\ell(\pi_j[\Delta_{ji}])).$$

Where the last equality follows from Corollary 52. Hence,

$$\phi(\Pi) = \sum_{j=1}^n \ell(\pi_j) = \sum_{i=1}^n \ell(\bar{\gamma}_i) + \sum_{i=1}^n \sum_{j \neq i} O(\ell(\pi_j[\Delta_{ji}])) = \phi(\bar{\Gamma}) + \sum_{i=1}^n \sum_{j \neq i} O(\ell(\pi_j[\Delta_{ji}])).$$

By definition of Δ_{ji} , $\Delta_{ji} \subseteq [j-1, j]$ and $\pi_j[\Delta_{ji}] \subseteq B_{z_i}$. Fix a point $x \in \mathbb{R}^2$. Consider a disk D of radius 4 centered at x . If $x \in B_z$ for some $z \in \mathbf{s} \cup \mathbf{f}$, then $C_z \subseteq D$. Since cores are pairwise-disjoint (cf Lemma 43(i)), D can contain at most 16 core disks and any $t \in [j-1, j]$ lies in $O(1)$ Δ_{ji} 's. Therefore,

$$\sum_{i \neq j} O(\ell(\pi_j[\Delta_{ji}])) = O(\ell(\pi_j[j-1, j])) = O(\ell(\bar{\gamma}_j)).$$

Finally, we obtain $\phi(\Pi) = \phi(\bar{\Gamma}) + \sum_{j=1}^n O(\phi(\bar{\gamma}_j)) = O(\phi(\bar{\Gamma}))$. \square

6.4.3 Running-time Analysis

The algorithm has three stages. In the first stage, we compute the free space \mathcal{F} with respect to one robot, which takes $O(m \log m)$ time, by computing the Voronoi of \mathcal{W} , see the algorithm of [Yap87], and see [BHKP12] for details. In the same stage, we compute

a set of shortest paths Γ for n discs, using the algorithm of [CW15], taking $O(mn \log m)$ time in total over all robots. Each path $\gamma_i \in \Gamma$ has complexity $O(m)$. In stage two of the algorithm, γ_i is modified to avoid the core of any occupied revolving area, increasing the complexity of each curve to $O(m+n)$. In stage three of the algorithm, the deformed paths $\bar{\gamma}_i$ are again edited to include retraction maps in which non-active robots may move within their revolving area. It suffices to bound the number of breakpoints in the final path π_j that correspond to retracted maps. Let ξ be such a breakpoint on π_j , which is $\rho_{ij}(\bar{\gamma}_i(t))$ for some $t \in [i-1, i]$. There are two cases: (i) the preimage of ξ on $\bar{\gamma}_i$ is a breakpoint of $\bar{\gamma}_i$, or (ii) $\|\xi - z_j\| = 2$ (i.e., $\bar{\gamma}_i$ forces R_j to move within the revolving area). We charge both of these breakpoints to $\bar{\gamma}_i$. Since the preimage of ξ lies in the buffer disk of R_j , using a packing argument similar to the proof of Lemma 53, we can show that $O(m+n)$ breakpoints are charged to $\bar{\gamma}_i$.

Theorem 54. *Let $\mathcal{J} = (\mathcal{W}, \mathbf{s}, \mathbf{f}, \mathcal{A})$ be an instance of optimal MRMP with revolving areas, and let m be the complexity of \mathcal{W} . If a feasible motion plan of \mathcal{J} exists then a path ensemble Π of cost $O(\phi^*(\mathcal{J}))$ can be computed in $O(n(m+n) \log m)$ time.*

We conclude this section by noting that since the ordering σ (of active robots) is arbitrary, the algorithm can be extended to an online setting where R_i and (s_i, f_i) are given in an online manner (as long as each s_i, f_i given satisfies the revolving area property). Our algorithm is $O(1)$ -competitive for this setting, i.e., the cost is $O(1)$ times the optimal cost of the offline problem.

6.5 Computing a Good Ordering

In the previous section, we proved that the total cost of the path ensemble Π is $O(1) \cdot \phi^*(\mathcal{J})$ irrespective of the order in which the robots moved. However, the order in which robots move has a significant impact on how the paths are edited in Stages (II) and (III). The increase in cost because of editing may vary between 0 and $O(nm)$ depending on the ordering (see [SH18] for a related argument). For a path ensemble Π computed by our

algorithm, let $\Delta\phi(\Pi) = \phi(\Pi) - \phi(\Gamma)$, which we refer to as the *marginal cost* of Π , where Γ is the path ensemble computed in Stage (I). For a permutation σ of $[n]$, let Π_σ be the path ensemble computed by the algorithm if robots were moved in the order determined by σ . Set $\Delta\phi(\sigma) := \Delta\phi(\Pi_\sigma)$. Finally, set $\Delta\phi^*(\mathcal{J}) = \min_\sigma \Delta\phi(\sigma)$, where the minimum is taken over all permutations of $[n]$.

Adapting the construction in [SH18], we can show that the problem of determining whether $\Delta\phi^*(\mathcal{J}) \leq L$, for some $L \geq 0$, is NP-hard. We present an approximation algorithm for computing a good ordering σ such that $\Delta\phi(\sigma) = O(\log n \log \log n) \Delta\phi^*(\mathcal{J})$.

Our main observation is that $\Delta\phi(\sigma)$, the marginal cost of an ordering σ , is decomposable, in the sense made precise below. For a pair $i \neq j$, we define $w_{ij} \geq 0$ to be the contribution of the pair R_i, R_j to the marginal cost of an ordering σ , assuming $i \prec_\sigma j$, i.e., how much the shortest path γ_i has to be modified because of γ_j and vice-versa assuming R_i is active before R_j . Note that if $i \prec_\sigma j$ then R_i (resp. R_j) is at f_i (resp. s_j) when R_j (resp. R_i) is active. There are two components of w_{ij}^σ :

1. R_i (resp. R_j) enters the core C_{s_j} (resp. C_{f_i}) in γ_i (resp. γ_j),
2. retraction motion of R_j (resp. R_i) when R_i (resp. R_j) enters the buffer disc B_{s_j} (resp. B_{f_i}).

Let ϕ_{ij} (resp. ϕ_{ji}) be the arc of ΔC_{s_j} (resp. ΔC_{f_i}) with which $\gamma_i \cap C_{s_j}$ (resp. $\gamma_j \cap C_{f_i}$) is replaced with. Then $\alpha_{ij} = \ell(\phi_{ij}) + \ell(\phi_{ji}) - \ell(\gamma_i \cap C_{s_j}) - \ell(\gamma_j \cap C_{f_i})$ is the contribution of (i) to w_{ij} . For (ii), we define $\rho_{ij}^<$ (resp. $\rho_{ij}^>$) be the retraction map of R_j because of R_i when R_i is active before (resp. after) R_j . Then $w_{ij} = \alpha_{ij} + \ell(\rho_{ij}^<(\bar{\gamma}_i)) + \ell(\rho_{ij}^>(\bar{\gamma}_j))$. From the previous two components, we have $\Delta\phi(\sigma) = \sum_{i,j:i \prec_\sigma j} w_{ij}$.

We now reduce the problem of computing an optimal ordering to instance of *weighted feedback-arc-set* (FAS) problem. Given a directed graph with weights on the edges, $G = (V, E)$, $w : E \rightarrow \mathbb{R}_{\geq 0}$, a feedback arc set F is a subset of edges of G whose removal makes G a directed acyclic graph. The weight of F , $w(F)$, is $\sum_{e \in F} w(e)$. The FAS problem asks to compute an FAS of the smallest weight. It is known to be NP-complete.

Given an MRMP-RA instance $\mathcal{J} = (\mathcal{W}, \mathbf{s}, \mathbf{f}, \mathcal{A})$, we first compute Γ as in stage (I) of the algorithm. Next, for each pair $i, j \in [n]$, we construct a directed graph as follows. $G = (V, E)$ is a complete directed graph with $V = [n]$, one representing each robot, $E = \{i \rightarrow j : 1 \leq i \neq j \leq n\}$, $w(i \rightarrow j) = w_{ij}$. It can be shown that each feedback arc set F of G induces an ordering σ_F on $[n]$, and vice versa. Furthermore, $w(F) = \Delta\phi(\sigma_F)$. Even et al. [ENSS98] have described a polynomial-time $O(\log n \log \log n)$ -approximation algorithm for the FAS problem. By applying their algorithm to G , we obtain the following.

Theorem 55. *Let $\mathcal{J} = (\mathcal{W}, \mathbf{s}, \mathbf{f}, \mathcal{A})$ be an instance of optimal MMP with revolving areas, and let m be the complexity of \mathcal{W} . Let the optimal order of execution of paths be σ^* . An ordering σ with $\Delta\phi(\sigma) = O(\log n \log \log n)\Delta\phi(\sigma^*)$ can be computed in polynomial time in n and m .*

6.6 Makespan

The sum of lengths is not the right objective function for time-sensitive applications, as it does not distinguish between whether the robots move simultaneously or sequentially. In this section, we focus on the question of *scheduling* two *unit square* robots: given two paths, we want to compute a parameterization such that the time the second robot arrives at its final position is minimized, or the min-makespan of the given Π . In this setting, we constrain each robot to move with speed $s \in [0, 1]$. For simplicity we do not impose any constraints on the acceleration of a robot, so the speed can change instantaneously. We first discuss known result for the scheduling problem.

Related Work. In the discrete setting, the scheduling problem, in which you are given a set of paths and asked to find a feasible schedule, is called the routing problem. More formally, a routing problem $Q = (G, \Pi, P)$ on graph G of n vertices consists of a set of N packets Π that are to be routed on their given paths P , and the goal is to determine a feasible schedule for the packets, minimizing the total amount of time steps. At each time step i , the packets are located at a vertex, and in the next time step can move to an open

(unoccupied) neighboring vertex. For a given set of paths, we also define the edge-congestion as the maximum number of packets that use an edge in G , and the node-congestion as the maximum number of packets that use a node in G , and the dilation as the maximum path length in P . Let C be the congestion and D be the dilation for a given instance. Additionally, for each edge, there is allowed a *queue*, where packets can wait to move from one vertex to the next. In a seminal result, it is proved that there exists a schedule using $O(C + D)$ time steps and constant size queues at every edge, through random injection times of packets [LMR94]. In another result, there is a randomized bufferless (queueless) routing algorithm with routing time $O((C + D) \log^3(n + N))$. The main component is constructing a bufferless algorithm by emulating a buffered algorithm. While a packet is being “buffered” it circulates in a local area of the graph. Typically in routing problems, the input paths are considered to be fixed, but in order to circumvent lower bounds and eliminate queues, in this work the input paths are edited [BMIM04]. There are multiple challenges extending these results to our setting: a queue in our setting means that multiple robots may be colliding, and the bufferless algorithm relies on packets being able to “swap” places, which in our setting corresponds to two robots moving along colliding paths.

For results in the continuous setting, if there are no obstacles and every robot is a constant distance away from each other, the work shows that you can fit a mesh grid over the robots and work in the discrete setting of a grid graph. This work relies on the fact that all grid spaces are free, which is not the case once obstacles are introduced cite: dmaine2019coordinated.

Continuous Setting. More formally, suppose we are given Π : two continuous polygonal paths $\pi_a, \pi_b : I \rightarrow \mathcal{W}$ from $\mathbf{s} = \{s_a, s_b\}$ to $\mathbf{f} = \{f_a, f_b\}$. Let $L_A = \ell\pi_A$ denote the L_2 length of path π_A (similarly define L_B). Since the paths are polygonal, we can write them as a sequence of straight line edges, let $\pi_A = (e_1, e_2, \dots, e_w)$ and $\pi_B = (g_1, g_2, \dots, g_z)$. We will work in the free configuration space diagram \mathcal{C} , which is the parameter space in which the x -coordinate corresponds to the point robot R_A is at along π_A (resp. the y -coordinate corresponds to the point robot R_B is at along π_B). We call a point $(x, y) \in \mathcal{C}$ a *configura-*

tion. To construct \mathcal{C} , construct a $L_A \times L_B$ rectangle. The row (resp. column) boundaries of the diagram are at values y (resp. x) such that $\pi_A(y)$ (resp. $\pi_B(x)$) is a vertex of the path. Each row corresponds to an edge e_i of path π_A with length equal to $\ell(e_i)$. Let the L_∞ distance be denoted by $\|\cdot\|_\infty$. Note that two rectangular robots do not collide as long as their L_∞ distance is at least 2.

$$\mathcal{C} = \{(p, q) \mid (p, q) \in [0, L_A] \times [0, L_B] \text{ s.t. } \|\pi_A(x) - \pi_B(y)\|_\infty \geq 2\}.$$

Let the x -coordinate (resp. y -coordinate) of a point be denoted p_x (resp. p_y).

Lemma 56. *A directed segment pq in \mathcal{C} corresponds to a feasible motion plan H from configuration p to q where $\text{makespan}(H) = \|p - q\|_\infty$*

Proof. Without loss of generality, we assume $p_x < q_x$ and $y(p) < y(q)$. Let $\delta x = q_x - p_x$ and $\delta y = q_y - p_y$, and assume $\delta y > \delta x$. Let p_A be the point $\pi_A(p_y)$, q_A be the point $\pi_A(q_y)$ and similarly let p_B be the point $\pi_B(p_x)$ and q_B be the point $\pi_B(q_x)$. We claim there is a feasible plan from (p_A, p_B) to (q_A, q_B) with makespan equal to $\|p - q\|_\infty$ such that A and B move at constant speeds of at most 1 along their respective paths P_A, P_B . We define the plan as follows. robot A moves at unit speed for $\|p - q\|_\infty = \delta y$ time (moving δy distance), and robot B moves δx distance at $\delta x/\delta y$ speed for δy time. After δy time, A and B lies at point p and B lies at point q along their paths. Since pq is a segment in \mathcal{C} , A and B this motion is feasible, and has makespan $\delta y = \|p - q\|_\infty$.

To find the optimal scheduling of Π , we first construct the configuration space in time $O(wz \log(wz))$ using the algorithm of Alt and Godau [AG95]. We next compute a minimum cost L_∞ path in \mathcal{C} in $O(t \log t)$ time [SR94] where t is the number of pairs of edges $e_i, g_j \in \pi_A, \pi_B$ such that the segments are within distance 2. In the worst case, $t = wz$. Each new edge of the path defines a change in speed for the robots, and Lemma 56 gives us the parameterization of the schedule. We obtain the following theorem.

Theorem 57. *Given two paths π_A, π_B we can find the optimal min-makespan scheduling of the paths, or report that no feasible schedule exists in $O(wz \log wz)$ time.*

6.7 Conclusion

In this work, we presented the first constant-factor approximation algorithm for computing a feasible weakly-monotone motion plan to minimize the sum of distances traveled. Additionally, the algorithm can be extended to an online setting where the polygonal environment is fixed, but the initial and final positions of the robots are specified in an online manner. On the hardness side, we prove that minimizing the total traveled distance, even with the restriction of a weakly-monotone motion plan, is APX-hard. For the makespan objective, we showed how to schedule movement along given paths for two robots to achieve the optimal makespan among all feasible schedules restricted to the given paths.

There are several interesting open questions. The first is whether the constant factor approximation presented in this work can be improved; another is whether there are instances in which the separation bounds for revolving areas are not required or can be tightened. There are other objectives to consider; instead of the sum of distances objective, one can consider the makespan (latest arrival time), where little is known even for a small number of discs in the presence of obstacles.

Chapter 7

Conclusion

We conclude by giving a high-level overview of our problems and results and then discuss open problems for future research. At a high level, this research has focused on three distinct geometric optimization problems, employing various strategies, assumptions, and algorithms to address the challenges they present. Each problem encompasses a high-dimensional domain: in the clustering problem, we are in a metric space with unbounded doubling dimension, In the redistricting problem, we are searching over the space of all balanced, compact, k -partitions, and in the motion planning problem, we have $2k$ degrees of freedom. By giving novel algorithms to solve problems in each domain, we have contributed to advancing efficient, fair, and high-quality solutions in the realm of geometric optimization.

For the trajectory clustering problem, we gave approximation algorithms for k -median clustering under the discrete Fréchet and Hausdorff distance. It would be interesting if we could extend our results to a non-constant number of cluster centers k or to other distance measures such as the continuous Fréchet distance or dynamic time warping (though this is not a metric and presents additional challenges). It would also be interesting to provide other characterizations on the metric space for the cluster centers (as alternatives to the notion of covering discussed in this) that can fit into the same efficient clustering algorithm sampling paradigm.

For the redistricting problem, we considered the new notion of local fairness. It would be interesting if we could give explicit approximation algorithms for finding locally fair partitions in two-dimensions or the planar graph setting. It would also be interesting to extend our work to a stochastic setting, where each person has a probabilistic score of voting for a certain outcome, and we need high confidence in the non-existence of a c -

deviating group. For computing compact redistricting plans, it would be interesting if we further explored the potential of the proposed cutting method, including how to incorporate additional constraints required for real redistricting applications.

Finally, for the motion planning problem, we presented algorithms for computing an approximately minimum distance plan when the initial and final configurations of robots satisfy some separation properties. It would be interesting to further characterize solutions when a small number of robots fail to exhibit the required separation. For the makespan objective, we showed that we could optimally schedule given paths to minimize makespan for two robots and compute an approximate schedule for three robots. It would be interesting to extend this approximation to k robots or to show how to compute paths that achieve the optimal makespan.

Overall, we propose that in the face of immense amounts of data and interesting problems, there is value in finding provable good solutions, and analyzing the worst-case outcomes of algorithmic schemes can alleviate potential unexpected bad outcomes. Moreover, as the use of algorithms to automate decision-making in many impactful domains increases, these problems and results become more vital.

Bibliography

- [ABC⁺17] H. Aziz, M. Brill, V. Conitzer, E. Elkind, R. Freeman, and T. Walsh. Justified representation in approval-based committee voting. *Social Choice and Welfare*, 48(2):461–485, 2017.
- [ABS10] Marcel R. Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and nonmetric distance measures. *ACM Trans. Alg.*, 6(4):59:1–59:26, 2010.
- [AdBHS15] Aviv Adler, Mark de Berg, Dan Halperin, and Kiril Solovey. Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Trans Autom. Sci. Eng.*, 12(4):1309–1317, 2015.
- [AEH⁺18] Haris Aziz, Edith Elkind, Shenwei Huang, Martin Lackner, Luis Sánchez Fernández, and Piotr Skowron. On the complexity of extended and proportional justified representation. In *AAAI*, pages 902–909, 2018.
- [AG95] Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.
- [AGHT22] Pankaj K Agarwal, Tzvika Geft, Dan Halperin, and Erin Taylor. Multi-robot motion planning for unit discs with revolving areas. In *33rd International Symposium on Algorithms and Computation (ISAAC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.
- [AKMT22] Pankaj K. Agarwal, Shao-Heng Ko, Kamesh Munagala, and Erin Taylor. Locally fair partitioning. In *AAAI*, pages 4752–4759, 2022.
- [AM10] Micah Altman and Michael McDonald. The promise and perils of computers in redistricting. *Duke Journal of Constitutional Law & Public Policy*, 5:69, 2010.
- [AM11] Micah Altman and Michael P McDonald. BARD: Better automated redistricting. *Journal of Statistical Software*, 42:1–28, 2011.
- [AP02] Pankaj K Agarwal and Cecilia M Procopiuc. Exact and approximation algorithms for clustering. *Algorithmica*, 33(2):201–226, 2002.
- [ARR98] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proc. ACM Symp. Th. Comput.*, volume 98, pages 106–113, 1998.
- [BBDC⁺07] Nicolas Basalto, Roberto Bellotti, Francesco De Carlo, Paolo Facchi, Ester Pantaleo, and Saverio Pascazio. Hausdorff clustering of financial time series. *Physica A: Statistical Mechanics Applications*, 379(2):635–644, 2007.

- [BCD⁺21] Josh Brunner, Lily Chung, Erik D. Demaine, Dylan H. Hendrickson, Adam Hesterberg, Adam Suhl, and Avi Zeff. 1 X 1 rush hour with fixed blocks is PSPACE-complete. In *10th International Conference on Fun with Algorithms*, volume 157, pages 7:1–7:14, 2021.
- [BdBB⁺22] Bahareh Banyassady, Mark de Berg, Karl Bringmann, Kevin Buchin, Henning Fernau, Dan Halperin, Irina Kostitsyna, Yoshio Okamoto, and Stijn Slot. Unlabeled Multi-Robot Motion Planning with Tighter Separation Bounds. In *38th International Symposium on Computational Geometry (SoCG)*, 2022.
- [BDG⁺19] Kevin Buchin, Anne Driemel, Joachim Gudmundsson, Michael Horton, Irina Kostitsyna, Maarten Löffler, and Martijn Struijs. Approximating (k,l)-center clustering for curves. In *Proc. ACM-SIAM Symp. Disc. Alg.*, pages 2922–2938. SIAM, 2019.
- [BDR23] Maïke Buchin, Anne Driemel, and Dennis Rohde. Approximating (k, l)-median clustering for polygonal curves. *ACM Transactions on Algorithms*, 19(1):1–32, 2023.
- [BDS19] Kevin Buchin, Anne Driemel, and Martijn Struijs. On the hardness of computing an average curve. *arXiv preprint arXiv:1902.08053*, 2019.
- [BGH⁺17] Sachet Bangia, Christy Vaughn Graves, Gregory Herschlag, Han Sung Kang, Justin Luo, Jonathan C Mattingly, and Robert Ravier. Redistricting: Drawing the line, 2017.
- [BGLR16] P. C. Besse, B. Guillouet, J. Loubes, and F. Royer. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 17(11):3306–3317, 2016.
- [BH17] Bethune-Hill. Bethune-Hill v. Virginia Board of Elections. https://www.supremecourt.gov/opinions/18pdf/18-281_6j37.pdf, 2017.
- [BHKP12] Eric Berberich, Dan Halperin, Michael Kerber, and Roza Pogalnikova. Deconstructing approximate offsets. *Discret. Comput. Geom.*, 48(4):964–989, 2012.
- [BHPI02] Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proc. ACM Symp. Th. Computing*, pages 250–257. ACM, 2002.
- [BJW⁺08] Sergey Bereg, Minghui Jiang, Wencheng Wang, Boting Yang, and Binhai Zhu. Simplifying 3D polygonal chains under the discrete Fréchet distance. In *Lat. Amer. Symp. Theoret. Informatics*, pages 630–641. Springer, 2008.
- [BKS07] Steven J. Brams, D. Marc Kilgour, and M. Remzi Sanver. A minimax procedure for electing committees. *Public Choice*, 132(3):401–420, 2007.
- [BLSS18] Allan Borodin, Omer Lev, Nisarg Shah, and Tyrone Strangway. Big city vs. the great outdoors: voter distribution and how it affects gerrymandering. In *IJCAI*, pages 98–104, 2018.

- [BMIM04] Costas Busch, Malik Magdon-Ismael, and Marios Mavronicolas. Universal bufferless routing. In *International Workshop on Approximation and Online Algorithms*, pages 239–252. Springer, 2004.
- [BS16] Arturs Backurs and Anastasios Sidiropoulos. Constant-distortion embeddings of Hausdorff metrics into constant-dimensional l_p spaces. In *APPROX/RANDOM*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
- [BS20] Amariah Becker and Justin Solomon. Redistricting algorithms, 2020.
- [CAKM19] Vincent Cohen-Addad, Philip N Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in Euclidean and minor-free metrics. *SIAM J. Comput.*, 48(2):644–667, 2019.
- [CAKY18] Vincent Cohen-Addad, Philip N Klein, and Neal E Young. Balanced centroidal power diagrams for redistricting. In *ACM SIGSPATIAL*, pages 389–396, 2018.
- [CC83] John R. Chamberlin and Paul N. Courant. Representative deliberations and representative decisions: Proportional representation and the borda rule. *The American Political Science Review*, 77(3):718–733, 1983.
- [CDO00] Carmen Cirincione, Thomas A Darling, and Timothy G O’Rourke. Assessing South Carolina’s 1990s congressional districting. *Political Geography*, 19(2):189–211, 2000.
- [CFLM19] Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. Proportionally fair clustering. In *ICML*, pages 1032–1041, 2019.
- [Che21] Matt Chen. Tufts research lab aids states with redistricting process. <https://tuftsdaily.com/news/2021/04/06/tufts-research-lab-aids-states-with-redistricting-process/>, 2021.
- [CKLV17] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. In *NeurIPS*, pages 5029–5037, 2017.
- [CKM⁺21] Vincent Cohen-Addad, Philip N. Klein, Dániel Marx, Archer Wheeler, and Christopher Wolfram. On the computational tractability of a geographic clustering problem arising in redistricting. In *FORC*, pages 3:1–3:18, 2021.
- [CLLV22] Moses Charikar, Paul Liu, Tianyu Liu, and Thuy-Duong Vuong. On the complexity of sampling redistricting plans, 2022.
- [Cool17] Cooper. Cooper v. Harris. <https://supreme.justia.com/cases/federal/us/581/15-1262/>, 2017.
- [CR13] Jowei Chen and Jonathan Rodden. Unintentional gerrymandering: Political geography and electoral bias in legislatures. *Quarterly Journal of Political Science*, 8(3):239–269, 2013.

- [CRSV22] Marion Campisi, Thomas Ratliff, Stephanie Somersille, and Ellen Veomett. Geography and election outcome metric: An introduction. *Election Law Journal: Rules, Politics, and Policy*, 2022.
- [CW15] Danny Z Chen and Haitao Wang. Computing shortest paths among curved obstacles in the plane. *ACM Transactions on Algorithms*, 11(4):1–46, 2015.
- [CWLS11] Jinyang Chen, Rangding Wang, Liangxu Liu, and Jiatao Song. Clustering of trajectories based on Hausdorff distance. In *Proc. Int. Conf. Electronics Comm. Control*, pages 1940–1944. IEEE, 2011.
- [DDS20] Daryl DeFord, Moon Duchin, and Justin Solomon. A computational approach to measuring vote elasticity and competitiveness. *Statistics and Public Policy*, 7(1):69–86, 2020.
- [DDS21a] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: a family of markov chains for redistricting. *Harvard Data Science Review*, 2021.
- [DDS21b] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: A family of markov chains for redistricting. *Harvard Data Science Review*, 3(1), 3 2021. <https://hdsr.mitpress.mit.edu/pub/1ds8ptxu>.
- [DFK⁺19] Erik D. Demaine, Sándor P. Fekete, Phillip Keldenich, Henk Meijer, and Christian Scheffer. Coordinated motion planning: Reconfiguring a swarm of labeled robots with bounded stretch. *SIAM Journal on Computing*, 48(6):1727–1762, 2019.
- [DKS16] Anne Driemel, Amer Krivošija, and Christian Sohler. Clustering time series under the Fréchet distance. In *Proc. ACM-SIAM Symp. Disc. Alg.*, pages 766–785. SIAM, 2016.
- [Dro81] H. R. Droop. On methods of electing representatives. *Journal of the Statistical Society of London*, 44(2):141–202, 1881.
- [DSPH21] Dror Dayan, Kiril Solovey, Marco Pavone, and Dan Halperin. Near-optimal multi-robot motion planning with finite sampling. In *IEEE International Conference on Robotics and Automation*, pages 9190–9196, 2021.
- [DT18] Moon Duchin and Bridget Eileen Tenner. Discrete geometry for electoral geography. *arXiv preprint arXiv:1808.05860*, 2018.
- [EGB22] Seyed A Esmaili, Hayley Grape, and Brian Brubach. Centralized fairness for redistricting. *arXiv preprint arXiv:2203.00872*, 2022.
- [EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. ACM Int. Conf. Know. Disc. Data Mining*, volume 96, pages 226–231, 1996.

- [EM94] Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical report, Information Systems Dept., Technical University of Vienna, 1994.
- [ENSS98] Guy Even, J Seffi Naor, Baruch Schieber, and Madhu Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998.
- [FG88] Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 434–444. ACM, 1988.
- [FHIT20] Benjamin Fifield, Michael Higgins, Kosuke Imai, and Alexander Tarr. Automated redistricting simulation using Markov chain Monte Carlo. *Journal of Computational and Graphical Statistics*, 29(4):715–728, 2020.
- [Fol70] Duncan K Foley. Lindahl’s solution and the core of an economy with public goods. *Econometrica*, pages 66–72, 1970.
- [GGRS22] Nikhil Garg, Wes Gurnee, David Rothschild, and David B. Shmoys. Combating gerrymandering with social choice: The design of multi-member districts. In *EC*, pages 560–561, 2022.
- [GGS94] Walter Gander, Gene H. Golub, and Rolf Strebler. Least-squares fitting of circles and ellipses. *Election Law Journal: Rules, Politics, and Policy*, 34(4):558–578, 1994.
- [GH21] Tzvika Geft and Dan Halperin. On the complexity of a family of decoupled multi-robot motion planning problems, 2021.
- [GH22] Tzvika Geft and Dan Halperin. Refined hardness of distance-optimal multi-agent path finding. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pages 481–488, 2022.
- [GI03] Venkatesan Guruswami and Piotr Indyk. Embeddings and non-approximability of geometric problems. In *Proc. ACM-SIAM Symp. Disc. Alg.*, volume 3, pages 537–538, 2003.
- [GKM⁺21] Paweł Gawrychowski, Haim Kaplan, Shay Mozes, Micha Sharir, and Oren Weimann. Voronoi diagrams on planar graphs, and computing the diameter in deterministic $\tilde{O}(n^{5/3})$ time. *SIAM Journal on Computing*, 50(2):509–554, 2021.
- [God14] Sebastian Goderbauer. Political districting for elections to the german bundestag: an optimization-based multi-stage heuristic respecting administrative boundaries. In *Operations Research*, pages 181–187, 2014.
- [GS62] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

- [GS99] Scott Gaffney and Padhraic Smyth. Trajectory clustering with mixtures of regression models. In *Proc. ACM Int. Conf. Know. Disc. Data Mining*, volume 99, pages 63–72, 1999.
- [GS21] Wes Gurnee and David B. Shmoys. Fairmandering: A column generation heuristic for fairness-optimized political districting. In *SIAM ACDA*, pages 88–99, 2021.
- [Hau78] Felix Hausdorff. *Grundzüge der Mengenlehre*, volume 61. American Mathematical Soc., 1978.
- [HKL⁺20] Gregory Herschlag, Han Sung Kang, Justin Luo, Christy Vaughn Graves, Sachet Bangia, Robert Ravier, and Jonathan C Mattingly. Quantifying gerrymandering in North Carolina. *Statistics and Public Policy*, 7(1):30–38, 2020.
- [HKR93] Daniel P Huttenlocher, Gregory A Klanderman, and William J Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, 15(9):850–863, 1993.
- [HP94] Dorit S Hochbaum and Anu Pathria. Node-optimal connected k-subgraphs. *University of California, Berkeley*, 1994.
- [HPK07] Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *Disc. Computat. Geom.*, 37(1):3–19, 2007.
- [HPL15] Chih-Chieh Hung, Wen-Chih Peng, and Wang-Chien Lee. Clustering and aggregating clues of trajectories for mining trajectory patterns and routes. *Int. J. Very Large Databases*, 24(2):169–192, 2015.
- [HPM04] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proc. ACM Symp. Th. Computing*, pages 291–300. ACM, 2004.
- [HRM17] Gregory Herschlag, Robert Ravier, and Jonathan C Mattingly. Evaluating partisan gerrymandering in Wisconsin. *arXiv preprint arXiv:1709.01596*, 2017.
- [HSS84] John E Hopcroft, Jacob Theodore Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "warehouseman's problem". *The International Journal of Robotics Research*, 3(4):76–88, 1984.
- [Hub23] Redistricting Data Hub. Redistricting data hub. <https://redistrictingdatahub.org/>, 2023.
- [HWS⁺65] Sidney Wayne Hess, JB Weaver, HJ Siegfeldt, JN Whelan, and PA Zitlau. Nonpartisan political redistricting by computer. *Operations Research*, 13(6):998–1006, 1965.

- [IKLZ17] Nicole Immorlica, Robert Kleinberg, Brendan Lucier, and Morteza Zadomighaddam. Exponential segregation in a two-dimensional schelling model with tolerant individuals. In *ACM-SIAM SODA*, pages 984–993, 2017.
- [JMS02] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proc. ACM Symp. Th. Computing*, pages 731–740. ACM, 2002.
- [KF11] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.
- [KJS15] Douglas M King, Sheldon H Jacobson, and Edward C Sewell. Efficient graph contiguity and hole algorithms for geographic zoning and dynamic plane graph partitioning. *Mathematical Programming*, 149(1):425–457, 2015.
- [KMV19] Richard Kueng, Dustin G. Mixon, and Soledad Villar. Fair redistricting is hard. *Theoretical Computer Science*, 791:28–35, 2019.
- [KR07] Stavros G Kolliopoulos and Satish Rao. A nearly linear-time approximation scheme for the Euclidean k-median problem. *SIAM J. Comput.*, 37(3):757–782, 2007.
- [KSS05] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear time algorithms for clustering problems in any dimensions. In *Int. Coll. Automata Lang. Programming*, pages 1374–1385. Springer, 2005.
- [KSS10] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2):5, 2010.
- [KTAM22] Shao-Heng Ko, Erin Taylor, Pankaj Agarwal, and Kamesh Munagala. All politics is local: Redistricting via local fairness. *Advances in Neural Information Processing Systems*, 35:17443–17455, 2022.
- [LCC⁺22] Jerry Lin, Carolyn Chen, Marc Chmielewski, Samia Zaman, and Brandon Fain. Auditing for gerrymandering by identifying disenfranchised individuals. In *2022 ACM FAccT*, pages 1125–1135, 2022.
- [LCW16a] Yan Y. Liu, Wendy K. Tam Cho, and Shaowen Wang. PEAR: a massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and Evolutionary Computation*, 30:78–92, 2016.
- [LCW16b] Yan Y. Liu, Wendy K. Tam Cho, and Shaowen Wang. PEAR: A massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and Evolutionary Computation*, 30:78–92, 2016.
- [Lev22] Justin Levitt. All about redistricting. <https://redistricting.lls.edu/>, 2022.
- [LF19] Harry A. Levin and Sorelle A. Friedler. Automated congressional redistricting. *ACM Journal of Experimental Algorithmics*, 24(1):1.10:1–1.10:24, 2019.

- [Lin58] Erik Lindahl. Just taxation—a positive solution. In *Classics in the Theory of Public Finance*, pages 168–176. Springer, 1958.
- [LMR94] Frank Thomson Leighton, Bruce M Maggs, and Satish B Rao. Packet routing and job-shop scheduling in (α, β) steps. *Combinatorica*, 14(2):167–186, 1994.
- [LS16] Shi Li and Ola Svensson. Approximating k -median via pseudo-approximation. *SIAM J. Computing*, 45(2):530–547, 2016.
- [Mat21] Jonathan Mattingly. Expert report on the north carolina state legislature and congressional redistricting (corrected version), 2021.
- [MGG22] MGGG. MGGG states. <https://github.com/mggg-states>, 2022.
- [Mil95] Miller. Miller v. Johnson. <https://supreme.justia.com/cases/federal/us/515/900/>, 1995.
- [MKS⁺22] Cory McCartan, Christopher T Kenny, Tyler Simko, George Garcia III, Kevin Wang, Melissa Wu, Shiro Kuriwaki, and Kosuke Imai. Simulated redistricting plans for the analysis and evaluation of redistricting in the united states. *Scientific Data*, 9(1):689, 2022.
- [MM18] Daniel B Magleby and Daniel B Mosesson. A new approach for developing neutral redistricting plans. *Political Analysis*, 26(2):147–167, 2018.
- [MMR19] Stefan Meintrup, Alexander Munteanu, and Dennis Rohde. Random projections and sampling algorithms for clustering of high-dimensional polygonal curves. *Advances in Neural Information Processing Systems*, 32, 2019.
- [Mon95] Burt L. Monroe. Fully proportional representation. *The American Political Science Review*, 89(4):925–940, 1995.
- [MS84] Nimrod Megiddo and Kenneth J Supowit. On the complexity of some common geometric location problems. *SIAM J. Comput.*, 13(1):182–196, 1984.
- [NT20] Abhinandan Nath and Erin Taylor. k -median clustering under discrete Fréchet and Hausdorff distances. In *36th International Symposium on Computational Geometry (SoCG 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
- [NT21] Abhinandan Nath and Erin Taylor. k -median clustering under discrete fréchet and hausdorff distances. *Journal of Computational Geometry*, 12(2):156–182, 2021.
- [OY85] C O’Dnlaing and CK Yap. A retraction method for planning the motion of a disc. *J. Algorithms*, 6:104–111, 1985.
- [PP91] Daniel D Polsby and Robert D Popper. The third criterion: Compactness as a procedural safeguard against partisan gerrymandering. *Yale Law & Policy Review*, 9:301, 1991.

- [Pro22] Princeton Gerrymandering Project. Princeton gerrymandering project. <https://gerrymander.princeton.edu/>, 2022.
- [PTF22] Ariel D Procaccia and Jamie Tucker-Foltz. Compact redistricting plans have many spanning trees. In *ACM-SIAM SODA*, pages 3754–3771, 2022.
- [QZC09] Lin Qu, Fan Zhou, and YW Chen. Trajectory classification based on Hausdorff distance for visual surveillance system. *J. Jilin University*, 6:1618–1624, 2009.
- [Reo61] Ernest C Reock. A note: Measuring compactness as a requirement of legislative apportionment. *Midwest Journal of Political Science*, 5(1):70–74, 1961.
- [Rey64] Reynolds. Reynolds v. Sims. <https://supreme.justia.com/cases/federal/us/377/533/>, 1964.
- [Sal19] Oren Salzman. Sampling-based robot motion planning. *Commun. ACM*, 62(10):54–63, 2019.
- [SBD07] Lukas Svec, Sam Burden, and Aaron Dilley. Applying voronoi diagrams to the redistricting problem. *The UMAP journal*, 28(3):313–329, 2007.
- [Sca67] Herbert E Scarf. The core of an n person game. *Econometrica: Journal of the Econometric Society*, pages 50–69, 1967.
- [Sch71] Thomas C. Schelling. Dynamic models of segregation. *The Journal of Mathematical Sociology*, 1(2):143–186, 1971.
- [SFEL⁺17] L. Sánchez-Fernández, E. Elkind, M. Lackner, N. Fernández, J. A. Fisteus, P. Basanta Val, and P. Skowron. Proportional justified representation. In *AAAI*, pages 670–676, 2017.
- [SFR12] Cynthia Sung, Dan Feldman, and Daniela Rus. Trajectory clustering for motion prediction. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1547–1552. IEEE, 2012.
- [SH16] Kiril Solovey and Dan Halperin. On the hardness of unlabeled multi-robot motion planning. *Int. J. Robotics Res.*, 35(14):1750–1759, 2016.
- [SH18] Israela Solomon and Dan Halperin. Motion planning for multiple unit-ball robots in \mathbb{R}^d . In *Workshop on the Algorithmic Foundations of Robotics, WAFR*, pages 799–816, 2018.
- [SJS⁺20] Kiril Solovey, Lucas Janson, Edward Schmerling, Emilio Frazzoli, and Marco Pavone. Revisiting the asymptotic optimality of RRT. In *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*, pages 2189–2195. IEEE, 2020.
- [SM15] Nicholas O Stephanopoulos and Eric M McGhee. Partisan gerrymandering and the efficiency gap. *University of Chicago Law Review*, 82:831, 2015.

- [SR94] James A Storer and John H Reif. Shortest paths in the plane with polygonal obstacles. *Journal of the ACM (JACM)*, 41(5):982–1012, 1994.
- [SS74] L. Shapley and H. Scarf. On cores and indivisibility. *Journal of Mathematical Economics*, 1(1):23–37, 1974.
- [SS83] Jacob T Schwartz and Micha Sharir. On the piano movers’ problem: III. coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers. *The International Journal of Robotics Research*, 2(3):46–75, 1983.
- [SSD⁺20] Rahul Shome, Kiril Solovey, Andrew Dobson, Dan Halperin, and Kostas E. Bekris. dRRT^{*}: Scalable and informed asymptotically-optimal multi-robot motion planning. *Auton. Robots*, 44(3-4):443–467, 2020.
- [SSF⁺19] Roni Stern, Nathan R. Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne T. Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, T. K. Satish Kumar, Roman Barták, and Eli Boyarski. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proc. 12th International Symposium on Combinatorial Search*, pages 151–159, 2019.
- [SYZH15] Kiril Solovey, Jingjin Yu, Or Zamir, and Dan Halperin. Motion planning for unlabeled discs with optimality guarantees. In *Robotics: Science and Systems*, 2015.
- [TCL16] Wendy K Tam Cho and Yan Y Liu. Toward a talismanic redistricting tool: A computational method for identifying extreme redistricting plans. *Election Law Journal*, 15(4):351–366, 2016.
- [TMMK14] Matthew Turpin, Kartik Mohta, Nathan Michael, and Vijay Kumar. Goal assignment and trajectory planning for large teams of interchangeable robots. *Auton. Robots*, 37(4):401–415, 2014.
- [Vaz01] Vijay V Vazirani. *Approximation algorithms*. Springer, 2001.
- [Vic61] William Vickrey. On the prevention of gerrymandering. *Political Science Quarterly*, 76(1):105–110, 1961.
- [Wan16] Samuel S-H Wang. Three tests for practical evaluation of partisan gerrymandering. *Stanford Law Review*, 68:1263, 2016.
- [War18] Gregory S Warrington. Quantifying gerrymandering using the vote distribution. *Election Law Journal*, 17(1):39–57, 2018.
- [WK20] Archer Wheeler and Philip N Klein. The impact of highly compact algorithmic redistricting on the rural-versus-urban balance. In *ACM SIGSPATIAL*, pages 397–400, 2020.
- [XZLZ15] Hongteng Xu, Yang Zhou, Weiyao Lin, and Hongyuan Zha. Unsupervised trajectory clustering via adaptive multi-kernel-based shrinkage. In *Proc. IEEE Int. Conf. Comp. Vision*, pages 4328–4336, 2015.

- [Yap87] Chee-Keng Yap. An $O(n \log n)$ algorithm for the voronoi diagram of a set of simple curve segments. *Discret. Comput. Geom.*, 2:365–393, 1987.
- [Zha11] Junfu Zhang. Tipping and residential segregation: a unified schelling model. *Journal of Regional Science*, 51(1):167–193, 2011.
- [ZVGRG17] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *ACM WWW*, pages 1171–1180, 2017.

Biography

Erin Taylor is a Ph.D. candidate in computer science at Duke University, advised by Pankaj Agarwal. She received her Bachelor of Science degree in computer science and mathematics from Duke University in 2018.