Algorithmica (2023) 85:965–975
https://doi.org/10.1007/s00453-022-01059-y

Few Cuts Meet Many Point Sets

Sariel Har-Peled[1] · **Mitchell Jones**[1]

Received: 19 May 2020 / Accepted: 2 November 2022 / Published online: 14 November 2022

Abstract
We study the problem of how to split many point sets in $\mathbb{R}^d$ into smaller parts using a few (shared) splitting hyperplanes. This problem is related to the classical Ham-Sandwich Theorem. We provide a logarithmic approximation to the optimal solution using the greedy algorithm for submodular optimization.

Keywords Ham-Sandwich Theorem · Submodular optimization · Approximation algorithms

1 Introduction

1.1 Motivation and the Problem

A basic problem in algorithms is partitioning the data effectively, so that one can apply divide and conquer algorithms. Recently, there was significant progress [1–3] on using polynomials to perform such partitions (e.g., polynomial Ham-Sandwich Theorem) to derive better combinatorial bounds (and in some cases, algorithms). Thus, polynomials provide a "universal" solution to this problem—however, there are some technical difficulties in handling polynomials efficiently. This work deals with alternative efficient partitioning geometric schemes using lines or hyperplanes.

Example: Separating points by a polynomial As a concrete example, consider the problem of splitting a point set $P \subset \mathbb{R}^2$ into singletons. This requires computing a non-zero polynomial $p(x, y)$, with a zero set $Z = \left\{(x, y) \in \mathbb{R}^2 \;\middle|\; p(x, y) = 0\right\}$, such that for every point of $P \setminus Z$ lies in its own connected component of $\mathbb{R}^2 \setminus Z$.

✉ Sariel Har-Peled
sariel@illinois.edu

Mitchell Jones
mitchell.jones1994@gmail.com

[1] Department of Computer Science, University of Illinois Urbana-Champaign, 201 N. Goodwin Avenue, Urbana, IL 61801, USA

Such a polynomial can be computed using the polynomial Ham-Sandwich theorem. At the ith stage, the point set is partitioned into 2^i sets $P_1^i, \ldots, P_{2^i}^i$ of similar cardinality. The idea is now to lift the points of P into 2^i dimensions. To this end, let $\mathcal{F}(i)$ be the set of the first i monomials over x and y ordered lexicographically by their degrees (i.e., $\mathcal{F}(i) = \{x, y, xy, x^2, y^2, xy, x^3, \ldots\}$). One then map a point $(x, y) \in \mathbb{R}^2$, to the corresponding point $(x, y, xy, x^2, y^2, x^3, \ldots)$ in 2^i dimensions, where each coordinate is a monomial from the set $\mathcal{F}(i)$. In the lifted space, one can now halve all 2^i sets by a single hyperplane, as guaranteed by the Ham-Sandwich Theorem, which in the original plane corresponds to a polynomial. This breaks P into 2^{i+1} sets, and one continues to the next iteration. If f_i is the polynomial computed in the ith iteration, for $i = 1, \ldots, h = \log n$, then the zero set Z_f of the product polynomial $f(x, y) = \prod_{i=1}^{h} f_i(x, y)$ breaks the plane into the desired components, as can be easily verified.

Why partitioning by polynomials is sometime not sufficient

The main issue is that the zero sets of polynomials are not easy to manipulate. If one preserves the representation of f as a product polynomial, as described above, then it is easy to decide if two points have the same sign pattern. Note that two points that have the same sign pattern might not be in the same connected component of $\mathbb{R}^2 \setminus Z_f$. In particular, deciding if two points are in the same connected component becomes much harder if the polynomial is not provided in this product form. Furthermore, this representation is not easy to modify and adapt (for example, modifying the representation if a few more points are inserted). As mentioned above, a natural alternative is to separate points by lines (or hyperplanes in higher dimensions). Here, two points p, q are separated by a given set of lines if there is at least one line in the set that intersects the interior of the segment pq.

The specific problem: Halving point sets The input is m sets $P_1, \ldots, P_m$ of points in $\mathbb{R}^d$, not necessarily disjoint (with $m > d$). Our goal is to split these sets into equal parts using a minimal number of hyperplanes. For $m \leq d$, the Ham-Sandwich Theorem states that one can bisect all of the sets using a single hyperplane. However, for $m > d$ and non-degenerate inputs, this is no longer possible. In particular, the number of point sets m might be significantly larger than d. As already demonstrated, one way to get around this restriction is via the polynomial Ham-Sandwich Theorem [4].

Here, we are interested in what can be done with restricted entities, such as (several) hyperplanes. To keep the problem feasible, we somewhat relax the problem—the requirement is no longer that each piece of P_i is exactly half the size of the original set, but rather that it is sufficiently small, as formally specified next.

Problem 1 Let $P_1, \ldots, P_m$ be $m > d$ point sets in $\mathbb{R}^d$, not necessarily disjoint, with $n = \sum_i |P_i|$. Let $\mu_1, \ldots, \mu_m$ be integers with $0 < \mu_i \leq |P_i|$. The goal is to compute the smallest set of hyperplanes H, such that for every cell ψ in the arrangement $\mathcal{A}(H)$ of hyperplanes, $|P_i \cap \psi| \leq \mu_i$ for all i. See Fig. 1 for an example.

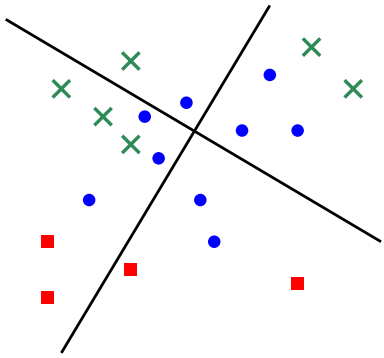

Fig. 1 Given three point sets, suppose the goal is to break the green (cross) point set into sets with at most three points, the blue (dot) point set into sets with at most four points, and the red (square) point set into sets with at most two points. This can be achieved using two separating lines (Color figure online)

This problem is interesting even for $d = 2, m = 1$, and $\mu_1 = 1$ – this is the problem of breaking a set of points in the plane into singletons using lines. Currently, only a logarithmic approximation is known [5].

Applications

One natural application of this problem comes from machine learning. Given a (single) point set P of size n in $\mathbb{R}^d$ and a collection of features $f_1, \ldots, f_m$, where $f_i : \mathbb{R}^d \to \mathbb{R}$, f_i *distinguishes* between two points p and q if $f_i(p)$ and $f_i(q)$ have different signs. Given a collection of features $S \subseteq \{f_1, \ldots, f_m\}$ one can assign each point p a vector,[1] $v_S(p) \in \{-1, 1\}^{|S|}$, where each entry of $v_S(p)$ is the sign of a feature in S evaluated at p. The point $v_S(p)$ is the *signature* of p with respect to S. Consider the task of choosing a subset of features $S \subseteq \{f_1, \ldots, f_m\}$, where $|S|$ is as small as possible, such that for any $u \in \{-1, 1\}^{|S|}$, the number of points with the same signature as u is at most $n/2$, formally $|\{p \in P \mid u = v_S(p)\}| \leq n/c$, with $c = 2$. The choice $c = 2$ in the last statement is arbitrary—but there is a tradeoff between the value of c and the size of S. For example, if we fix the size of S, then $c = 2^{|S|}$ is not feasible for $|S| \gg d$, when using hyperplanes, as an arrangement of $|S|$ hyperplanes in $\mathbb{R}^d$ has only $\tau = O(|S|^d) \ll 2^{|S|}$ different cells (of various dimensions). The later implies that one cell of the arrangement must be assigned at least $n/\tau > n/2^{|S|} = n/c$ points of P.

Furthermore, one would like to apply this to several point sets $P_1, \ldots, P_\ell$, so one can select the smallest number of features (i.e., $|S|$) such that for all $u \in \{-1, 1\}^{|S|}$, $|\{p \in P_i \mid u = v_S(p)\}| \leq |P_i|/2$, for all $i = 1, \ldots, \ell$. A natural scenario for such an application is in the realm of big data. Given a collection of (large) data sets, it needs to be divided among different computers. The fewer the features needed to get a split as described above, the faster one can decide where to send each input point. Here, the required guarantee is that each set gets reduced to at most half its size.

Thus, once one decides which features to use, one can scan the data and compute for each input point its signature, which determines where it is being stored. This naturally can be done in a distributed fashion, so that each computer/node has to compute the signature only for a small portion of the data.

[1] For the sake simplicity of exposition, we ignore here the case that the sign is zero.

For the case where all the points have to be singletons in the induced partition of features, this can be interpreted as a non-linear dimension reduction of the input set into a hypercube, where the dimension of the hypercube is as small as possible. Indeed, once we picked a set of s hyperplanes, $h_1, \ldots, h_s$, each one of them has an associated sign function $f_i(p) \in \{-1, 1\}$, where a point p (not lying on any of the planes) has $f_i(x) = 1$ if p is on one side of h_i, and -1 if p is on the other size. This naturally defines an embedding of P to the hypercube $\{-1, 1\}^s$, as for all $p \in P$, we have $F(p) = \big(f_1(p), \ldots, f_s(p)\big) \in \{0, 1\}^s$.

1.2 Background

Ham sandwich theorem The Ham-Sandwich Theorem is a well studied problem in both mathematics and computer science. Since its inception, there have been many results related to computing such cuts in higher dimensions [6], as well as generalizations of the theorem [4, 7–10]. For example, one such generalization is the following: Given well separated convex bodies $C_1, \ldots, C_d$ in $\mathbb{R}^d$ and constants $\mu_i \in [0, 1]$, there exists a unique hyperplane h that contains at least a μ_i fraction of the volume on the positive side h^+ for $i = 1, \ldots, d$ [7]. This result was then extended to discrete point sets under certain conditions [11]. Notably, in this paper we consider the case when the number of point sets can be much larger than the ambient dimension d. The problem of simultaneously bisecting more than d convex bodies in $\mathbb{R}^d$ using multiple hyperplanes has been studied combinatorially [8, 10], whereas our focus is on the algorithmic aspects.

Other generalizations include the polynomial Ham-Sandwich Theorem, in which one is interested in partitioning a point set using polynomials rather than hyperplanes [4, 12]. This generalization, and the original Ham-Sandwich Theorem has a variety of applications in geometric range searching [13, 14].

Partial set cover

An instance of the *set cover* problem is a pair (G, Π), where G is the *ground set*, and $\Pi \subseteq 2^G$ is set of (hyper)edges. The problem is to compute edges $f_1, \ldots, f_t \in \Pi$, such that $\cup_i f_i = G$. Usually, one wants to minimize t—the number of edges used.

In the partial set cover problem, one is interested in covering at least a certain number of the elements in a set system (this number is the *demand* of this instance), using as few sets as possible. Specifically, an instance of this problem is a tuple (G, Π, d) (first two parameters are as in the set cover problem, and $\mathsf{d} \geq 0$), and the problem is to compute edges $f_1, \ldots, f_t \in \Pi$, such that $\left|\cup_{i=1}^{t} f_i\right| \geq \mathsf{d}$ (again, the target is usually to minimize t). For our purposes, we need a parallel version of this problem. This variant is formally defined in Problem 5 below. In this parallel version, there are many set systems sharing sets, each with its own demand, where the demand is the minimal number of elements that have to be covered in each instance.

For the standard partial set cover problem, an $O(\log n)$-approximation is well known, and follows from the greedy algorithm (see below for details). In geometric settings, Inamdar and Varadarajan [15] showed that partial set cover can be approximated to within $O(\beta)$, where β is the approximation ratio for the set cover version of the problem. Because many geometric problems admit much better than $O(\log n)$-

approximations, this results in an improvement to the partial set cover version of the problem. However, it is not clear how to apply their algorithm in the parallel setting.

1.3 Our Results

We reduce Problem 1 to a generalized instance of *partial set cover*, where we allow multiple ground sets, with different demands, and show that the standard greedy algorithm for submodular optimization can be applied to this problem.

Sketch of the greedy algorithm To solve Problem 1, let $U = \cup_i P_i$ and let H be the collection of all combinatorially different hyperplanes with respect to U. Consider the arrangement $\mathcal{A} = \mathcal{A}(H)$ of H. We introduce an edge between a pair of points of P if they lie in the same cell of $\mathcal{A}$. If we consider the process of adding the hyperplanes from H as an incremental process, then initially every point is in the same cell as all the other points. Modeling this as a graph, we start with a clique, and every hyperplane h added disconnects the edges which correspond to segments that h intersects. In particular, a point is in a cell with at most m points if it has degree $m - 1$ in the remaining graph. As such, this can be interpreted as a parallel version of set cover, where every vertex induces its own instance, which requires a certain number of edges adjacent to it to be covered (i.e., cut). Naturally, parallel versions of set cover can be solved using a greedy algorithm that picks the hyperplane that cuts the largest number of edges that still need cutting (being somewhat informal). However, it is somewhat more natural to describe the greedy algorithm using the framework of submodular optimization.

Paper organization In Sect. 2 we provide the necessary background on minimization under submodular constraint needed for our main result. We then show how to solve the multiple partial set cover problem in Sect. 3.1. Next, in Sect. 3.2, we study the problem of partitioning a set into smaller sets, such that each element in each of the smaller sets meet a given demand requirement. The final result, stated in Theorem 15, provides a logarithmic approximation for our problem by reducing it to the aforementioned problems.

2 Preliminaries

For a set X, and an element x, let $X + x = X \cup \{x\}$, and $X - x = X \setminus \{x\}$. A *set system* is a pair (G, Π), with $\Pi \subseteq 2^G$. The set system (G, Π) can also be viewed as a hypergraph with the vertex set G, and the sets in Π as (hyper) ***edges*** .

2.1 Submodular Minimization

For the sake of completeness, we describe the greedy algorithm for finding a minimal solution satisfying an integer valued submodular constraint. In this case, the task is to compute the smallest set of edges that provides the same utility as using all the edges available.

Let (G, Π) be a given set system, and assume we have a monotone (non-negative) function $f : 2^{\Pi} \to \mathbb{Z}$. Here a function is ***monotone*** if $\mathcal{Z} \subseteq \mathcal{Y} \subseteq \Pi$ implies that $f(\mathcal{Z}) \leq f(\mathcal{Y}) \leq f(\Pi)$. Intuitively, the function $f(\mathcal{Z})$ measures the *benefit* of a set $\mathcal{Z}$ – the higher the value of f is, the higher the benefit. In particular, $f_{\max} = f(\Pi)$ is the maximum benefit possible.

We also assume that f is *submodular*, that is for any $e \in \Pi$, and for all $\mathcal{Z} \subseteq \mathcal{Y} \subseteq \Pi \setminus \{e\}$, we have that

$$\Delta_{\mathcal{Z}}(e) = f(\mathcal{Z} + e) - f(\mathcal{Z}) \geq f(\mathcal{Y} + e) - f(\mathcal{Y}) = \Delta_{\mathcal{Y}}(e).$$

Submodularity is known in economics as *diminishing returns*– the marginal benefit (per unit) of allocating more resources to solve a problem decreases as more resources are allocated.

Problem 2 Under the above settings, the problem at hand is to compute (or approximate) the smallest (cardinality) set $\mathcal{O} \subseteq \Pi$, such that $f(\mathcal{O}) = f_{\max}$.

Example 3 Consider an instance of set cover (G, Π), with $n = |G|$. Given a family $\mathcal{Z} \subseteq \Pi$ of edges, its *benefit* is the number of elements in G the edges of $\mathcal{Z}$ cover. That is, $f(\mathcal{Z}) = \left|\cup_{\varkappa \in \mathcal{Z}} \varkappa\right|$. It is not hard to verify that f is monotone and submodular. Solving Problem 2 here corresponds to computing a minimum set cover for G.

Consider the greedy algorithm that starts with an empty solution $\mathcal{C}_0$. In the ith iteration, the algorithm picks the edge $e'_i \in \Pi$ that maximizes the value $f(\mathcal{C}_{i-1} + e'_i) - f(\mathcal{C}_{i-1})$, and updates $\mathcal{C}_i = \mathcal{C}_{i-1} + e'_i$. The algorithm stops when $f(\mathcal{C}_i) = f_{\max} = f(\Pi)$.

Theorem 4 [Wolsey [16]] *Given a set system (G, Π), and a non-negative monotone submodular function $f : 2^{\Pi} \to \mathbb{Z}$, the greedy algorithm, described above, outputs a solution with $O(k \log f_{\max})$ edges of Π, where $k = |\mathcal{O}|$ is the size of the smallest set $\mathcal{O} \subseteq \Pi$ such that $f(\mathcal{O}) = f_{\max} = f(\Pi)$.*

For the sake of completeness we include the proof in "Appendix A".

3 Problems and Reductions

3.1 PCMS: Partial Cover for Multiple Sets

Problem 5 [PCMS] The input is a set system (U, Π), and a collection $\mathcal{G} = \{G_i \subseteq U \mid i = 1, \ldots, m\}$ of ground sets, where the universe U is of size n. In addition, each ground set G_i has a ***demand*** , denoted by $\mathsf{d}(G_i)$, which is a non-negative integer. A valid solution for such an instance, is a collection $\mathcal{Y} \subseteq \Pi$, such that $\bigcup_{y \in \mathcal{Y}} y$ covers at least $\mathsf{d}(G_i)$ elements of G_i, for $i = 1, \ldots, m$.

Remark 6 In the following, to simplify the exposition, we assume that the given instances being solved are feasible. Otherwise, the approximation algorithm would fail to generate a solution thus proving the unfeasibility of the given instance.

Lemma 7 *Let* $(U, \mathcal{G}, \Pi)$ *be an instance of* partial cover of multiple sets *(PCMS), where* $n = |U|$, $\mathcal{G}$ *is a family of* m *ground sets, and* Π *is a family of edges. Furthermore, each ground set of* $\mathcal{G}$ *has an associated demand. Then, the greedy algorithm computes, in polynomial time, an* $O\big(\log(mn)\big)$*-approximation to the minimal size set* $\mathcal{O} \subseteq \Pi$ *that meets all the demands of the ground sets.*

Proof Consider a partial solution $\mathcal{C} \subseteq \Pi$. The *service* of $\mathcal{C}$ to G_i is

$$f_i(\mathcal{C}) = \min\big(|G_i \cap (\cup \mathcal{C})|\,,\ \mathsf{d}(G_i)\big),$$

where $\cup \mathcal{C} = \cup_{e \in \mathcal{C}} e$. That is, as long as the demand of $\mathsf{d}(G_i)$ is not met, $f_i(\mathcal{C})$ is the number of elements of G_i the union of the edges of $\mathcal{C}$ covers. Once the demand is met, f_i is maxed out at $\mathsf{d}(G_i)$. Observe that $f_i(\emptyset) = 0$, f_i is clearly monotone, and its maximal value is $\mathsf{d}(G_i) \leq n$. As for submodularity, consider sets $\mathcal{Z} \subseteq \mathcal{Y} \subseteq \Pi$, and an edge $e \in \Pi$, and note that $f_i(\mathcal{Z} + e) - f_i(\mathcal{Z}) \geq f_i(\mathcal{Y} + e) - f_i(\mathcal{Y})$, as e potentially covers more new elements of G_i when added to a smaller cover. For the given PCMS instance and a solution $\mathcal{Z} \subseteq \Pi$, the target function is

$$f(\mathcal{Z}) = \sum_{i=1}^{m} f_i(\mathcal{Z}).$$

The function f is a sum of submodular functions. As such, f is submodular itself. Observe that $f(\Pi) \leq mn$. Now, using the algorithm of Theorem 4 implies the result. □

Remark 8 One can obtain an $O(\log m)$-approximation for Problem 5 via LP rounding [17], which is useful when m is much smaller than n. However, for our main application (see Lemma 12), this does not change our final result as the number of ground sets is polynomial in n (i.e., $m = \Theta(n^2)$ in our case).

3.2 Cutting a Set into Smaller Pieces

We are given a set-system (G, Π), where $n = |G|$. A set $\mathcal{Z} \subseteq \Pi$ of edges, induces a natural partition of G, where two elements $x, y \in G$ are in the same set of the partition $\iff$ x and y belong to the same set of edges in $\mathcal{Z}$. Formally, $x \equiv y \iff \mathcal{Z} \cap x = \mathcal{Z} \cap y$, where $\mathcal{Z} \cap x = \{f \in \mathcal{Z} \mid x \in f\}$. The partition of G induced by $\mathcal{Z}$ (i.e., the equivalence classes of $\equiv$) is the ***arrangement*** of $\mathcal{Z}$, denoted by $\mathcal{A}(\mathcal{Z})$. A set of $\mathcal{A}(\mathcal{Z})$ is a ***face*** of $\mathcal{A}(\mathcal{Z})$. For an element $x \in G$, the face of $\mathcal{A}(\mathcal{Z})$ that contains x is denoted by $\mathrm{face}(x, \mathcal{Z})$.

Example 9 For $G = \{1, 2, 3, 4, 5\}$, and $\mathcal{Z} = \big\{\{1, 2, 3\}, \{3, 4, 5\}\big\}$, we have

$$\mathcal{A}(\mathcal{Z}) = \big\{\{1, 2\}, \{3\}, \{4, 5\}\big\}.$$

Problem 10 [Reduce by half] Given a set system (G, Π), with $n = |G|$, find a minimum sized set $\mathcal{Z} \subseteq \Pi$ such that every face of $A(\mathcal{Z})$ is of size at most $n/2$.

Problem 11 [PTD: Partition to demand] Given a set system (G, Π), where $n = |G|$, and an integral ***demand*** $\mathsf{d}(v) \geq 0$, for each $v \in G$, find a minimum sized set $\mathcal{Z} \subseteq \Pi$, such that for every $v \in G$, $|\mathrm{face}(v, \mathcal{Z})| \leq \mathsf{d}(v)$.

Observe that Problem 10 can be reduced to Problem 11 by setting the demand of every vertex in the ground set to $n/2$.

Lemma 12 *Given an instance (G, Π) of PTD, with $n = |G|$, there is a greedy algorithm that computes, in polynomial time, an $O(\log n)$-approximation to the optimal solution.*

Proof Consider the complete graph $K_n = (G, E)$, where $E = \{xy \mid x, y \in G\}$. For every element $x \in G$, consider the associated cut $E_x = \{xy \mid y \in G - x\}$. A set $e \in \Pi$ *cuts* xy if $|e \cap \{x, y\}| = 1$. In particular, let $\mathrm{cut}(e) = \{xy \mid x \in e, y \in G \setminus e\}$ be the set of edges of K_n that e cuts.

Now, a set of edges $\mathcal{Y} \subseteq \Pi$ meets the demand of $v \in G$, if the edges of $\mathcal{Y}$ cut at least $n - \mathsf{d}(v)$ edges of E_v (e.g., if $\mathsf{d}(v) = n - 1$, then one needs to cut one edge attached to v). Put differently, the partial cover $\bigcup_{e \in \mathcal{Y}} \mathrm{cut}(e)$ covers at least $n - \mathsf{d}(v)$ edges of E_v. Thus, let $U' = E$ be the universe set, and $\mathcal{G}' = \{E_v \mid v \in G\}$ be the set of ground sets. Here a ground set $E_v \in \mathcal{G}'$ has demand $\mathsf{d}(E_v) = n - \mathsf{d}(v)$. The family of allowable sets to be used in the cover is $\Pi' = \{\mathrm{cut}(e) \mid e \in \Pi\}$.

The triple $\left(U', \mathcal{G}', \Pi'\right)$ is an instance of PCMS, with $n' = \left|U'\right| = O(n^2)$ and $m' = \left|\mathcal{G}'\right| = n$. The greedy algorithm yields an $O\left(\log(n'm')\right)$-approximation in this case, by Lemma 7. As $\log(n'm') = O(\log n)$, the claim follows. □

3.3 Cutting a Ham-Sandwich into Small Pieces

Problem 13 [RMC: Reduce measures via cuts] The input is a triplet $(U, \mathcal{G}, \Pi)$ with $n = |U|$. Here $\mathcal{G} = \{G_i \subseteq U \mid i = 1, \ldots, m\}$ is a collection of ground sets that are not necessarily disjoint, and $\Pi \subseteq 2^U$ is a collection of edges. For every ground set G_i, there is an associated target size $\mu_i \leq |G_i|$. The problem is to compute a minimal set $\mathcal{O} \subseteq \Pi$, such that, for all i, and any face ψ of $\mathcal{A}(\mathcal{O})$, we have $|\psi \cap G_i| \leq \mu_i$.

Lemma 14 *Given a feasible instance $(U, \mathcal{G}, \Pi)$ of RMC with $n = |U|$ and $m = |\mathcal{G}|$, one can compute, in polynomial time, an $O(\log(nm))$-approximation to the smallest set $\mathcal{O} \subseteq \Pi$ that satisfies the given instance.*

Proof For a set $G_i \in \mathcal{G}$, and an element $v \in U$, let $\mathsf{d}_i(v) = \mu_i$, if $v \in G_i$, and otherwise $\mathsf{d}_i(v) = n$. The pair (U, Π) with the demand function $\mathsf{d}_i(\cdot)$ form an instance of PTD (Problem 11), and its approximation algorithm (see Lemma 12) has an associated submodular function $f_i(\cdot)$, that is non-negative, monotone, submodular and has maximum value at most n^2.

Consider the submodular function $f = \sum_i f_i$, and let $f_{\max} = f(\Pi)$. Clearly, f is submodular, monotone, and has maximum value at most mn^2. Furthermore, a subset $\mathcal{Y} \subseteq \Pi$ such that $f(\mathcal{Y}) = f_{\max}$ is a valid solution to the given instance. As such, one can plug this into the algorithm of Theorem 4 and get the desired approximation. □

With all of the ingredients assembled, we are ready to tackle Problem 1.

Theorem 15 *Let $P_1, \ldots, P_m$ be m (not necessarily disjoint) point sets in $\mathbb{R}^d$, where $n = \sum_i |P_i|$. For each point set P_i, we are given an integer parameter $0 < \mu_i \leq |P_i|$. The task at hand is to compute a minimal set of hyperplanes H such that for every face ψ in the arrangement $\mathcal{A}(H)$, ψ contains at most μ_i points of P_i, for all $i = 1, \ldots, m$. One can $O\big(\log(mn)\big)$-approximate, in $O(mn^{d+3})$ time, the optimal solution.*

Proof The reduction is straightforward and uses Lemma 14. Let the shared ground set be $U = \cup_i P_i$. Let $\mathcal{G}$ be the family of ground sets $\{G_i = P_i \mid i = 1, \ldots, m\}$. Finally, let H be the (finite) number of combinatorially different hyperplanes with respect to U. For each $h \in H$, let h^+ be one of the two halfspaces bounded by h (which halfspace is not important – taking the other one corresponds to "flipping" the corresponding coordinate of the signature induced the arrangement). Add the set $\left\{ p \in U \;\middle|\; p \in h^+ \right\}$ to the collection of subsets Π. The values μ_i remain unchanged. This forms an instance of Problem 13, and thus we can apply Lemma 14 to obtain the desired separating hyperplanes.

As for the running time, computing the set system takes $O(n^{d+2})$ time by brute force. Indeed, unraveling the above reduction, the shared ground set is made of $\binom{n}{2}$ pairs of points of U. Every point has up to m different sets of such pairs that needs to be partially covered. Fortunately, there are only $O(n^d)$ edges in the resulting set system. Evaluating the contribution of a new edge (in the set system) to the target function takes $O(n^2 m)$ time. As there are $O(n^d)$ edges in set system, it follows that evaluating all edges takes $O(n^{d+2}m)$ time. Finally, it is easy to verify that the algorithm performs at most n iterations. □

No effort was made to improve the running time of the algorithm of Theorem 15.

4 Open Problems

The most natural open problem is to improve the approximation quality of Theorem 15. The same applies to all the other problems here, which potentially might have better approximation ratios because of the underlying geometry. On the other hand, it would be interesting to prove (conditional) lower bounds on the hardness of approximation of these problems.

Acknowledgements Sariel Har-Peled was partially supported by NSF AF awards CCF-1421231, CCF-1217462, and CCF-1907400. Mitchell Jones was partially supported by NSF AF awards CCF-1421231 and CCF-1907400. The authors also thank the anonymous referees for their detailed and useful feedback.

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

Appendix A Proof of Theorem 4

Proof This result is by now classical, and we include the proof only for the sake of completeness. Let $\mathcal{O} = \{o_1, \ldots, o_k\}$ be the optimal solution. Consider a current

solution $\mathcal{C}_i \subseteq \Pi$ at iteration i, and observe that by monotonicity, we have

$$f_{\max} = f(\mathcal{O}) \leq f(\mathcal{C}_i \cup \mathcal{O}) \leq f_{\max}.$$

As such, we have $f(\mathcal{C}_i \cup \mathcal{O}) = f_{\max}$. Let $\Delta_i = f(\mathcal{O}) - f(\mathcal{C}_i)$ be the ***deficiency*** of $\mathcal{C}_i$. For $j = 0, \ldots, k$, let $\mathcal{S}_j = \mathcal{C}_i \cup \{o_1, \ldots, o_j\}$. Set $\delta_j = f(\mathcal{S}_j) - f(\mathcal{S}_{j-1})$. We have that

$$\sum_{j=1}^{k} \delta_j = f(\mathcal{C}_i \cup \mathcal{O}) - f(\mathcal{C}_i) = f_{\max} - f(\mathcal{C}_i) = \Delta_i.$$

Hence, there is an index j, such that $\delta_j \geq \Delta_i/k$. Now, by submodularity, we have that

$$f(\mathcal{C}_i + o_j) - f(\mathcal{C}_i) \geq f(\mathcal{S}_{j-1} + o_j) - f(\mathcal{S}_{j-1}) = \delta_j \geq \Delta_i/k.$$

However, the greedy algorithm adds an element e that maximizes the value of $\Delta_{\mathcal{C}_i}(e)$, which is at least Δ_i/k. Put differently, the added element decreases the deficiency of the current solution by a factor $\leq 1 - 1/k$. Therefore the deficiency in the end of the ith iteration is at most $\Delta_i \leq (1 - 1/k)^i \Delta_0 = (1 - 1/k)^i f(\mathcal{O})$. This quantity is less than one for $i = O(k \log f_{\max})$. □

References

1. Matousek, J., Patáková, Z.: Multilevel polynomial partitions and simplified range searching. Disc. Comput. Geom. **54**(1), 22–41 (2015). https://doi.org/10.1007/s00454-015-9701-2
2. Agarwal, P.K., Aronov, B., Ezra, E., Zahl, J.: Efficient algorithm for generalized polynomial partitioning and its applications. SIAM J. Comput. **50**(2), 760–787 (2021). https://doi.org/10.1137/19M1268550
3. Sheffer, A.: Polynomial Methods and Incidence Theory. Cambridge University Press, Cambridge (2022). https://doi.org/10.1017/9781108959988
4. Stone, A.H., Tukey, J.W.: Generalized "sandwich" theorems. Duke Math. J. **9**(2), 356–359 (1942). https://doi.org/10.1215/S0012-7094-42-00925-6
5. Har-Peled, S., Jones, M.: On separating points by lines. Disc. Comput. Geom. **63**(3), 705–730 (2020). https://doi.org/10.1007/s00454-019-00103-z
6. Lo, C.-Y., Matoušek, J., Steiger, W.: Algorithms for ham-sandwich cuts. Disc. Comput. Geom. **11**, 433–452 (1994). https://doi.org/10.1007/BF02574017
7. Bárány, I., Hubard, A., Jerónimo, J.: Slicing convex sets and measures by a hyperplane. Disc. Comput. Geom. **39**(1–3), 67–75 (2008). https://doi.org/10.1007/s00454-007-9021-2
8. Blagojević, P.V., Soberón, P.: Thieves can make sandwiches. Bull. Lond. Math. Soc. **50**(1), 108–123 (2018). https://doi.org/10.1112/blms.12109
9. Ramos, E.A.: Equipartition of mass distributions by hyperplanes. Disc. Comput. Geom. **15**(2), 147–167 (1996). https://doi.org/10.1007/BF02717729
10. Schnider, P.: Ham-sandwich cuts and center transversals in subspaces. In: Proc. 35th Int. Annu. Sympos. Comput. Geom. (SoCG). LIPIcs, vol. 129, pp. 56– 15615 (2019). https://doi.org/10.4230/LIPIcs.SoCG.2019.56
11. Steiger, W., Zhao, J.: Generalized ham-sandwich cuts. Disc. Comput. Geom. **44**(3), 535–545 (2010). https://doi.org/10.1007/s00454-009-9225-8
12. Kaplan, H., Matoušek, J., Sharir, M.: Simple proofs of classical theorems in discrete geometry via the Guth-Katz polynomial partitioning technique. Disc. Comput. Geom. **48**(3), 499–517 (2012). https://doi.org/10.1007/s00454-012-9443-3

13. Agarwal, P.K., Matoušek, J., Sharir, M.: On range searching with semialgebraic sets. II. SIAM J. Comput. **42**(6), 2039–2062 (2013). https://doi.org/10.1137/120890855
14. Matoušek, J.: Geometric range searching. ACM Comput. Surv. **26**(4), 421–461 (1994). https://doi.org/10.1145/197405.197408
15. Inamdar, T., Varadarajan, K.R.: On partial covering for geometric set systems. In: Speckmann, B., Tóth, C.D. (eds.) Proc. 34th Int. Annu. Sympos. Comput. Geom. (SoCG). LIPIcs, vol. 99, pp. 47–14714. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Wadern, Germany (2018). https://doi.org/10.4230/LIPIcs.SoCG.2018.47
16. Wolsey, L.A.: An analysis of the greedy algorithm for the submodular set covering problem. Combinatorica **2**(4), 385–393 (1982). https://doi.org/10.1007/BF02579435
17. Kolliopoulos, S.G., Young, N.E.: Approximation algorithms for covering/packing integer programs. J. Comput. Syst. Sci. **71**(4), 495–505 (2005). https://doi.org/10.1016/j.jcss.2005.05.002