Geometric Optimization Revisited

Pankaj K. Agarwal[1], Esther Ezra[2,3], and Kyle Fox[4(✉)]

[1] Department of Computer Science, Duke University, Durham, NC 27708-0129, USA
[2] Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel
[3] School of Math, Georgia Institute of Technology, Atlanta, GA 30332, USA
[4] Department of Computer Science, The University of Texas at Dallas, Richardson, TX 75080, USA
kyle.fox@utdallas.edu

Abstract. Many combinatorial optimization problems such as set cover, clustering, and graph matching have been formulated in geometric settings. We review the progress made in recent years on a number of such geometric optimization problems, with an emphasis on how geometry has been exploited to develop better algorithms. Instead of discussing many problems, we focus on a few problems, namely, set cover, hitting set, independent set, and computing maps between point sets.

1 Introduction

A combinatorial optimization problem is often formulated as maximizing or minimizing a function of one or more variables subject to a number of inequality (and equality) constraints. In contrast to continuous optimization, the set of feasible solutions is discrete or can be reduced to discrete. Because several problems in a wide range of fields such as mathematics, operations research, statistics, graph algorithms, machine learning, auction theory, robotics, GIS, computer graphics, and computational geometry can be formulated as combinatorial optimization problems, it has been an active research area for more than seventy years. Many of the combinatorial optimization problems are known to be NP-Hard and thus much attention has focused on fast approximation algorithms. With the increasing size of data sets, faster, say, near-linear-time, approximation algorithms are needed even if polynomial-time exact algorithms exist.

In many applications, the underlying optimization problem involves a large number of constraints that are induced by a set of geometric objects in low dimensions. In fact, one of the earliest combinatorial optimization problems studied was the transportation-plan problem in Euclidean space (see Sect. 4 below for a formal definition), which was first investigated by the French mathematician

P.K. Agarwal and K. Fox are supported in part by NSF under grants CCF-15-13816, CCF-15-46392, and IIS-14-08846, by ARO grant W911NF-15-1-0408, and by grant 2012/229 from the U.S.-Israel Binational Science Foundation. E. Ezra is supported in part by NSF CAREER under grant CCF-15-53354, and by Grant 824/17 from the Israel Science Foundation.

© Springer Nature Switzerland AG 2019
B. Steffen and G. Woeginger (Eds.): Computing and Software Science, LNCS 10000, pp. 66–84, 2019.
https://doi.org/10.1007/978-3-319-91908-9_5

Gaspard Monge in 1784. Furthermore, many classical combinatorial problems have been formulated in geometric settings; e.g., the Euclidean traveling salesperson problem: computing the shortest tour in a set of points in $\mathbb{R}^d$; the k-center problem: covering a set of points by k congruent disks of the smallest radius; and the geometric hitting-set problem: stabbing a set of simple geometric objects (e.g. disks) by the minimum number of points. We refer to such combinatorial optimization problems as *geometric optimization* problems.

A natural question in the context of geometric optimization problems is whether geometry can be exploited to obtain faster algorithms or to improve the approximation factor in the case of approximation algorithms. This question has received much attention over the last forty years and tremendous progress has been made on a large class of problems. For example, faster algorithms have been developed for low-dimensional linear programming [49], and a polynomial-time ε-approximation algorithm is now known for the Euclidean traveling salesperson problem [22,61] while no such algorithm exists for general graphs unless $\mathrm{P} = \mathrm{NP}$.

A survey paper by Agarwal and Sharir [14] in Vol. 1000 of LNCS (see [15] for an expanded version of this survey) reviews the work on geometric optimization until the early 1990s, including parametric search, prune-and-search, and randomized algorithms for LP-type problems. In the current volume, we discuss some of the main developments in this area over the last two decades, focusing on a few classical combinatorial optimization problems in geometric settings.

The first problem that we discuss (Sect. 2) is the classical set-cover or hitting-set problem in geometric settings, i.e., we have a set system whose elements are simple geometric objects such as points, lines, squares, disks, etc., and each set is a subset of geometric objects that satisfies certain geometric constraints (e.g. input points lying inside a disk). It turns out that both the set-cover and the hitting-set problems are NP-Complete even in very simple geometric settings. Therefore, attention has focused on developing fast approximation algorithms with as good an approximation factor as possible, including obtaining a trade-off between the running time and approximation factor. The work on geometric set-cover has also led to some interesting work in combinatorial geometry, including work on ε-nets and discrepancy theory.

A problem closely related to set-cover is the independent-set problem. Although the independent-set problem is intractable even from the approximation point of view for arbitrary graphs [76], it is easier to approximate in many geometric settings. Several interesting results have been obtained, including a quasi-polynomial ε-approximation algorithm that has found applications in many related problems (Sect. 3).

Next, we turn our attention in Sect. 4 to the problem of computing maps between two point sets to identify shared structures between them. We first consider the transportation-map problem mentioned earlier. Next, we discuss the problem of computing maps between two point sequences, which model time-series data or trajectories, and review recent results. We note that polynomial-time algorithms exist for both of these problems even in non-geometric settings.

2 Geometric Set Cover

A *set system* (or range space) $(\mathsf{P}, \mathcal{R})$ is a pair consisting of an underlying universe P of *objects* (also called the *space*) and a family $\mathcal{R}$ of subsets of P, referred to as *ranges*. A *set cover* of $(\mathsf{P}, \mathcal{R})$ is a subcollection $\mathcal{S} \subseteq \mathcal{R}$ such that $\bigcup \mathcal{S} = \mathsf{P}$. In a typical geometric setting, P is a set of points in $\mathbb{R}^d$, and each set in $\mathcal{R}$ is the intersection of P with a simply-shaped region (e.g., halfspaces, balls, simplices, etc.); with a slight abuse of notation, we will use $\mathcal{R}$ to denote the set of these regions as well. Given this setting, the geometric *set-cover* problem is to find a smallest subcollection of regions that cover the underlying set of points. The dual problem of set-cover is the *hitting-set* problem: find the smallest subset $H \subseteq \mathsf{P}$ such that $H \cap R \neq \emptyset$ for all $R \in \mathcal{R}$. The problem of set-cover (resp., hitting-set) can also be formulated where one assigns (non-negative) weights on the regions (resp., points), in which case the objective is to find a coverage (resp., hitting set) of smallest weight.

These problems were among the original NP-Complete problems [56], and the classic greedy approach leads to a $(1+\ln n)$-approximation algorithm, where $n = |\mathsf{P}|$ [74]. Dinur and Steurer [40] showed that a polynomial-time $o(\log n)$-approximation algorithm is infeasible unless P = NP. They remain NP-Complete even in simple 2D geometric settings such as when P is a set of points and $\mathcal{R}$ is a set of unit squares or disks [52, Chap. 8]. Nevertheless, the approximation factors achieved in geometric setups are considerably better than those obtained in abstract settings. We present an overview of these results in Sect. 2.2.

2.1 Greedy Algorithms

Let $(\mathsf{P}, \mathcal{R})$ be a geometric set system, where $\mathsf{P} \subseteq \mathbb{R}^d$ and $\mathcal{R}$ is a set of simply-shaped regions in $\mathbb{R}^d$. For the time being we assume P is finite, and set $n = |\mathsf{P}|$, $m = |\mathcal{R}|$. As mentioned above, the standard greedy approach yields a $(1+\ln n)$-approximation factor, but a naïve implementation of this approach is in general inefficient. By exploiting geometric properties of the set system, considerably faster algorithms can be obtained in many cases. We briefly present such an approach, introduced in [7], when $\mathsf{P} \subset \mathbb{R}^2$ and $\mathcal{R}$ is a collection of planar regions with low "union complexity", that is, the number of vertices and arcs appearing along the boundary of $\bigcup \mathcal{S}$, for any subset $\mathcal{S} \subseteq \mathcal{R}$, is close to linear. This is, e.g., the case for (pseudo-)disks[1] and "fat triangles"[2]; see [19] and the references therein. Here, we focus on the hitting-set problem.

Let Opt be the size of the optimal hitting set of $(\mathsf{P}, \mathcal{R})$. The main idea is to construct a set Π of $O(\mathsf{Opt})$ pairwise-disjoint cells (appearing in a plane decomposition induced by a subset of the regions in $\mathcal{R}$) that cover P. Each cell of Π has small complexity (e.g., cells are trapezoids or triangles), and each cell meets only a small fraction of the boundary regions in $\mathcal{R}$. Using the fact that

[1] Each of the regions is bounded by a closed Jordan curve, where each pair of boundary curves meet at most twice.

[2] In each triangle, every angle is greater than some constant.

the union complexity of $\mathcal{R}$ is close to linear, Agarwal *et al.* [7] show that such a Π can be constructed in near-linear time. They choose one point of P from each cell of Π, add it to the hitting set, remove the regions of $\mathcal{R}$ hit by the chosen points, and recurse. They show that the procedure terminates after $O(\log n)$ steps, thereby constructing a hitting set of size $O(\mathsf{Opt} \log n)$ in near-linear time.

Greedy algorithms have also been used to approximate the solution of the *art-gallery* problem, that is, the set-up is a polygonal domain P and the goal is to place a smallest number of guards in P, such that any point $p \in P$ is *visible* to at least one of these guards [70]. This problem is an instance of geometric set-cover in which the set $\mathsf{P} = P$ is infinite and so is the set of ranges, one range for each point $p \in P$ (namely, the *visibility polygon* of P). The art-gallery problem is known to be APX-Hard [43], and if the guard locations are chosen anywhere in the polygon, no polynomial-time approximation algorithm is known. In such cases, however, one can still obtain efficient approximation algorithms if they are allowed to leave a relatively small portion of P uncovered [6,37]. Very recently, Bonnet and Miltzow [26] showed an $O(\log \mathsf{Opt})$ approximation algorithm assuming the vertices of P have integer coordinates (as well as several general position assumptions). Moreover, they showed in a slightly earlier work [27] that when the guards are selected among the vertices of P (with arbitrary real coordinates), the *exact* problem cannot be solved in time $n^{o(\mathsf{Opt}/\log \mathsf{Opt})}$, unless the *Exponential Time Hypothesis (ETH)* fails.

2.2 Iterative Reweighing Scheme and ε-Nets

Not only can one obtain fast $O(\log n)$-approximation algorithms for hitting-set (or set-cover) in geometric settings, the approximation factor can be improved to $o(\log n)$. One general technique to obtain improved approximations is based on linear programming. Roughly speaking, one (implicitly) formulates the hitting-set problem as a linear program and uses an iterative reweighing scheme, proposed for solving packing and covering linear programs [44], to compute a near-optimal fractional solution. Then, the concept of ε-nets is used to construct an integral solution. We first describe ε-nets and then make the connection to set-cover and hitting-set.

ε-Nets. Let $\Sigma = (\mathsf{P}, \mathcal{R})$ be a (discrete) set system, and let $w : \mathsf{P} \to \mathbb{R}_{\geq 0}$ be a weight function. Given a parameter $0 < \varepsilon < 1$, a range $R \in \mathcal{R}$ is called *ε-heavy* if $w(R) \geq \varepsilon w(\mathsf{P})$ and *ε-light* otherwise. A subset $N \subseteq \mathsf{P}$ is called an *ε-net* of $(\mathsf{P}, \mathcal{R})$ if $N \cap R \neq \emptyset$ for every ε-heavy range $R \in \mathcal{R}$. If we flip the roles of P and $\mathcal{R}$, we obtain the so called *dual set system*: $\Sigma^* = (\mathcal{R}, \mathsf{P}^*)$, where $\mathsf{P}^* = \{\{R \in \mathcal{R} \mid p \in R\} \mid p \in \mathsf{P}\}$. We note that a set cover of Σ is a hitting set of Σ^* and vice-versa. In the dual case, an ε-net for $(\mathcal{R}, \mathsf{P}^*)$ is a subset $\mathcal{S} \subseteq \mathcal{R}$ that covers all the ε-heavy points of P, that is, those points that are originally covered by regions whose total weight is at least $\varepsilon w(\mathcal{R})$. Originally introduced as a tool for range-searching data structures in computational geometry, ε-nets have found many applications in computational and combinatorial geometry as well as in learning theory.

By a seminal result of Haussler and Welzl [49, Chap. 5], a geometric set system $(\mathsf{P}, \mathcal{R})$ has an ε-net of size $O(\varepsilon^{-1} \log \varepsilon^{-1})$ (the same asymptotic bound holds in dual set systems). In fact, a random sample of that size is an ε-net with constant probability. More generally, set systems of *VC-dimension* d admit ε-nets of size $O((d/\varepsilon) \log(d/\varepsilon))$; informally, this fact implies that the number of sets in $\mathcal{R}$ is polynomial in $|\mathsf{P}|$ (concretely, $O(|\mathsf{P}|^d)$), and that this property is hereditary for any restriction of the set system onto a subset of P [49, Chap. 5]. In general, it is known that the bound on ε-nets for set systems of finite VC-dimension is tight in the worst case [49, Chap. 5], and recently Pach and Tardos [65] showed that this bound is tight in several geometric settings. Still, there are several favorable geometric scenarios where this bound is smaller, that is, it is very close to $O(1/\varepsilon)$. For example, ε-nets of size $O(1/\varepsilon)$ exist when $\mathcal{R}$ is a set of halfspaces in two or three dimensions, pseudo-disks in the plane, or translates of a fixed convex polytope in 3-space; see, e.g., the references in [20]. The case of axis-parallel boxes in two and three dimensions was addressed by Aronov *et al.* [20], who showed an ε-net bound of $O(\varepsilon^{-1} \log\log \varepsilon^{-1})$; this bound has later been shown to be tight [65]. Additional progress has been made by Clarkson and Varadarajan [39], who introduced a method for constructing small-size ε-nets in several dual set systems. Specifically, they addressed the case where the underlying regions in $\mathcal{R}$ have low union complexity and showed that the resulting ε-nets have complexity $o(\varepsilon^{-1} \log \varepsilon^{-1})$. Later, this bound has further been improved by Aronov *et al.* [20]. For example, for fat triangles the bound is $O(\varepsilon^{-1} \log\log^* \varepsilon^{-1})$, and for more general fat objects the bound becomes $O(\varepsilon^{-1} \log^* \varepsilon^{-1})$.

Iterative Reweighing Scheme. We now describe a simple algorithm, which computes a hitting set of Σ using ε-nets [49, Chap. 6]; the algorithm can compute a set cover of Σ by running it on the dual range space Σ^*. Let k be an integer such that $k/2 < \mathsf{Opt} \leq k$. We initialize the weight of each point in P to 1 and repeat the following step until every range is $1/(2k)$-heavy. Let R be a $1/(2k)$-light range. We double the weights of all points in R. We refer to this step as a *weight-doubling* step. When the process stops, we return a $1/(2k)$-net H of Σ (with respect to the final weights), which serves as a hitting set (as at this point all ranges are $1/(2k)$-heavy). If a $1/(2k)$-net of size $O(kg(k))$ can be computed efficiently, we obtain a hitting set of size $O(\mathsf{Opt} \cdot g(\mathsf{Opt}))$.

Using a double-counting argument, it can be shown that if $\mathsf{Opt} \leq k$, then the above procedure terminates within $\mu(k) = O(k \log n)$ weight-doubling steps. Therefore, if the algorithm does not terminate within $\mu(k)$ steps, we can conclude that Σ does not have a hitting set of size at most k. An exponential search can then be used to guess the value of k such that $k/2 < \mathsf{Opt} \leq k$. Many methods have been proposed to check whether a $1/(2k)$-light range exists at each stage. For example, a $1/(2k)$-net can be computed at each step to detect a light range. Agarwal and Pan [12] described a data structure to detect light ranges, which leads to a near-linear-time implementation of the above algorithm in many cases. Putting these results together, one obtains a near-linear-time $O(1)$-approximation algorithm when ranges are disks in the plane or halfspaces

in $\mathbb{R}^3$, and an $O(\log\log \mathsf{Opt})$-approximation algorithm when ranges are rectangles or fat triangles. See [12] for details and concrete results.

As noted above, this algorithm is nothing but the iterative-reweighing algorithm proposed by packing and covering linear programs. The set of final weights that the algorithm computes is a near-optimal fractional solution, and the ε-net is being used to "round" this fractional solution into an integral solution. Agarwal and Pan presented a simpler iterative-reweighing scheme for computing a hitting set or set cover by formulating the problem as 2-player zero-sum game and computing a near-optimal strategy for each of the two players. The strategy for one of the players gives a near-optimal fractional solution for the hitting set, and then they use ε-nets to compute the actual hitting set [12].

2.3 Extensions

Weighted Set-Cover. The above iterative-reweighing technique is in general not extendible to compute a hitting set (or set cover) of smallest weight. Specifically, an $O(\log \mathsf{Opt})$-approximation factor can be obtained when all weights are greater than 1 [44], but the situation is unclear for arbitrary weight functions. Moreover, one may encounter the issue that this latter approximation factor can be arbitrarily larger than $\log n$ (e.g., if the input weights are large), which defeats the advantage geometric settings may have over abstract ones.

Notable progress on this problem was made by Varadarajan [72], who used a *quasi-uniform sampling* approach in order to find an approximately optimal weighted set cover, where the regions in $\mathcal{R}$ have low union complexity. The resulting approximation factors almost match those in the unweighted case. Chan *et al.* [34] later strengthened Varadarajan's technique and obtained asymptotically matching bounds.

Multi-Set-Cover. Another problem of interest is that of *multi-set-cover*, that is, in addition to the set system $(\mathsf{P}, \mathcal{R})$, each point $p \in \mathsf{P}$ has an integer demand $\mathsf{d}(p)$, and the goal is to find a smallest set cover so that each p is covered by $\mathsf{d}(p)$ (distinct) regions of the cover. For abstract set systems, the standard greedy approach results in a $(1 + \ln n)$-approximation factor [74], but the LP-relaxation approach based on ε-nets does not extend to the multi-cover setting. This problem was studied by Chekuri *et al.* [36], who showed similar asymptotic approximation factors as those obtained for the (standard) geometric set-cover. Their proof technique is somewhat intricate and involves a reduction to set cover with objects of one dimensional higher, which is the key to obtain an $O(\log \mathsf{Opt})$-approximation.

PTAS: Shifted Grids and Local Search. Notwithstanding that polynomial-time approximation schemes (PTASes) for the hitting-set/set-cover problem exist in a few cases, including pseudo-disks in the plane and halfspaces in 3-space [63], a PTAS is much more difficult to obtain in general. For instance, even for the cases of fat triangles in the plane (of similar size with angles close to 60 degrees) and fat axis-parallel rectangles (of similar size), the set-cover problem is APX-Hard

[33,51]. In fact, the work in [51] presents several such hardness results, where the overall conclusion is that for "non-shallow" (and even very simple) geometric settings, the hitting-set/set-cover problem tends to be APX-Hard, whereas in *shallow* settings it is more likely to have a PTAS[3].

For the case of points and unit disks in the plane, if only the points are given but the disks can be drawn anywhere in the plane, then one can obtain a PTAS for set cover using a "sliding grid" technique due to Hochbaum and Maas [52, Chap. 9]. The main observation is that in an optimal solution any given point is covered by only a constant number of disks. Therefore, any grid cell in the plane with constant side length must contain only a few disks in the optimal solution. This immediately yields a PTAS for hitting-set of points and unit disks by observing that a disk D covers a point p if and only if the unit disk centered at p is hit by the center of D. Chan [31] addressed the case of disks with arbitrary radii, and presented a PTAS for hitting-set based on *planar separators*. If the disk set is discrete, the problem becomes considerably more difficult; Mustafa and Ray [63] presented a PTAS for hitting set of arbitrary disks exploiting a local-search algorithm. In a more recent paper, Mustafa *et al.* [62] studied the weighted set cover problem for arbitrary disks, where they obtained a quasi-polynomial-time approximation scheme (QPTAS); their technique was inspired by a result of Adamaszek and Wiese [2] for approximating the maximum independent set for axis-parallel rectangles; we review this result is detail in Sect. 3.

3 Geometric Independent Set

Given a collection $\mathcal{R}$ of n (simply-shaped) regions, the geometric *maximum independent-set* problem is to find a maximum-cardinality subset $S \subset \mathcal{R}$ of pairwise disjoint regions. This is the largest independent set in the *intersection graph* of $\mathcal{R}$, and for brevity we just refer to the problem as the *independent-set* problem. When the regions in $\mathcal{R}$ are assigned (non-negative) weights, the *maximum-weight independent-set* problem is to find a maximum-weight subset $S \subseteq \mathcal{R}$ of pairwise-disjoint regions.

In abstract graphs, the independent-set problem is known to be NP-Complete. Unless NP = ZPP, no polynomial-time algorithm with approximation factor better than $n^{1-\varepsilon}$, for any $\varepsilon > 0$, exists [52, Chap. 10], and the currently best known approximation is $O(n(\log\log n)^2/(\log n)^3)$ [45]. Moreover, even when the maximum degree of the graph is 3, a PTAS does not exist. Recently Har-Peled and Quanrud [51] showed that it is possible to construct intersection graphs of triangles in 3-space for which a PTAS does not exist. However, better approximation algorithms have been developed for some geometric settings, and in some cases even a PTAS exists, e.g., the cases of fat objects [31] (such as disks and squares) and pseudo-disks [35].

Of particular interest is the case of axis-parallel rectangles which attracted much attention over the past decade [2,30,35,38] because of its application to

[3] A collection of geometric objects is *shallow* if any point in the ambient space is covered by only a few such objects.

"label placement" in a map—place on top of a given map as many labels as possible from a given collection without conflicts; each label is represented as a rectangle, and the rectangles are required to be pairwise disjoint. Nevertheless, there is also a high theoretical interest in this setting since the currently known PTAS techniques for fat objects do not extend to axis-parallel rectangles. In the sequel we describe these techniques and their consequences for different cases.

PTAS for Fat Objects. We begin with reviewing the case of fat objects. First, the case of unit disks is handled using the "sliding grid" technique [52, Chap. 9]. As in set-cover, the main observation is that a grid cell of side-length Δ can contain only $O(\Delta^2)$ pairwise-disjoint unit disks in an optimal solution (but unlike set-cover, the disks here are *given*). If the disks have arbitrary radii, the grid is replaced by a hierarchical spatial subdivision. Specifically, Chan [31] used *randomly shifted quadtrees*, originally proposed for Euclidean TSP [22]. His construction extends to any dimension $d \geq 2$; moreover, this extension applies to any set of fat objects, as the packing argument applied for (unit) disks still holds. An interesting consequence of Chan's algorithm is that it extends to the weighted case as well.

PTAS for Pseudo-Disks. The techniques for fat objects do not extend to pseudo-disks, as they strongly rely on a packing argument exploiting the fatness property. For the unweighted case, Chan and Har-Peled [35] showed a PTAS using a simple local-search approach and exploiting a planar separator theorem. An interesting consequence of their analysis is that a local-search algorithm yields a PTAS for the independent-set problem in planar graphs, giving an alternative approach to existing algorithms [76, Chap. 10]. In the weighted case, Chan and Har-Peled [35] used a clever LP-relaxation approach, where the main idea is to sparsify the intersection graph (which is a consequence of the low union complexity of pseudo-disks), and then apply *Turán's theorem* in order to extract the desired independent set. The latter implies that in every graph on n vertices and average degree d, there is an independent set of size at least $n/(d+1)$.

Axis-Parallel Rectangles. The main difficulty in obtaining small approximation factors for non-fat objects (excluding pseudo-disks) is the fact that their union complexity is typically large, e.g., for axis-parallel rectangles in the plane this complexity is quadratic (realized by a set of long and skinny rectangles forming a grid). It is fairly standard to obtain a logarithmic approximation for axis-parallel rectangles in the plane, using, e.g., a naïve LP-relaxation. However, improving the approximation factor further is considerably more challenging. Chan and Har-Peled [35] showed an $O(\log n/\log\log n)$-approximation for the weighted case, and an intricate LP-relaxation based approach by Chalermsook and Chuzhoy [30] obtains an $O(\log\log n)$-approximation for the unweighted case, which is currently the best known. Although an $O(1)$-approximation algorithm for axis-parallel rectangles has remained elusive, the recent QPTAS for the weighted case by Adamaszek and Wiese [2] is a major step forward. They present a $(1-\varepsilon)$-approximation algorithm that runs in $n^{O(\mathrm{poly}(\log n)/\varepsilon)}$ time. A key idea in their analysis is a new geometric dynamic programming scheme, which

recursively subdivides the plane into polygons of bounded complexity. Chuzhoy and Ene [38] obtained an improved QPTAS for the unweighted case, reducing the running time to $n^{O(\mathrm{poly}(\log\log n)/\varepsilon)}$. These results suggest that the problem is unlikely to be APX-Hard.

Non-rectilinear Non-fat Objects. For arbitrarily oriented non-fat objects such as segments or triangles, the approximation factors are worse, because the structural properties imposed by such objects tend to be weaker. Among these settings is the case of line-segments in the plane, studied by Agarwal and Mustafa [11], who obtained an approximation factor close to $O(\sqrt{\mathsf{Opt}})$ using properties of partially ordered sets. Fox and Pach [46] improved and generalized this bound to curves, each pair of which have a bounded number of intersections. In this case a QPTAS was shown, using the existence of separators in intersection graphs. The case of arbitrary polygons in the plane has been studied by Har-Peled [50], who showed a QPTAS based on the decomposition in [2].

4 Maps Between Point Sets

In this section, we consider the problem of computing a map between two point sets, say, A and B, each representing a geometric object, with the goal of identifying common structure (similarity) between them. This is a central problem in shape analysis, which has been studied extensively in computational geometry, computer vision, computer graphics, molecular biology, and machine learning. For simplicity, we assume A and B to be finite point sets, and in some cases a weight may be associated with each point. We seek a *correspondence* $C \subseteq A \times B$ subject to some restriction on C, e.g., each point of A and B should appear in at least one pair in C or there is some penalty for each point not included in any pair of C. A cost function is defined to measure how well C measures the common structure between A and B, and the goal is to compute a correspondence that minimizes the cost function.

The cost function generally falls under one of two loosely defined categories. *Extrinsic* cost functions are based on the embedding of the points in the ambient space, while *intrinsic* cost functions are based on properties of the objects represented by A and B and not on their embedding in the ambient space. Examples of the former include the Hausdorff distance and the Fréchet distance, and an example of the latter is the Gromov-Hausdorff distance. Extrinsic cost functions are generally easier to optimize using combinatorial algorithms, and many fast algorithms exist; see [17] for a survey of earlier results. Intrinsic cost functions are significantly more difficult and most of the known algorithms rely on numerical methods; see [54] for a survey.

In this section, we will focus on two settings of extrinsic cost functions: transportation maps between two weighted point sets and order preserving maps between point sequences. We will then briefly discuss their extensions and the relatively sparse landscape of combinatorial optimization for intrinsic measures.

4.1 Transportation Maps

Let R and B be two point sets in $\mathbb{R}^d$, and let $|R| = |B| = n$. Let $\lambda : R \cup B \to \mathbb{N}$ denote the *supplies* of points in A and the *demands* of points in B. We assume $\sum_{r \in R} \lambda(r) = \sum_{b \in B} \lambda(b) = U$. A function $\tau : R \times B \to \mathbb{N}$ is a *transportation map* between R and B if $\sum_{b \in B} \tau(r, b) = \lambda(r)$ for all $r \in R$ and $\sum_{r \in R} \tau(r, b) = \lambda(b)$ for all $b \in B$.

Let $\mathrm{d}(\cdot, \cdot)$ be a suitable distance function. We define the cost of a transportation map τ to be $\mu(\tau) = \sum_{(r,b) \in R \times B} \mathrm{d}(r, b)\tau(r, b)$. The *Hitchcock-Koopmans transportation problem* (often referred to simply as the *transportation problem*) is to find a minimum-cost transportation map τ^*. The cost $\mu(\tau^*)$ is called the *transportation distance* or *earth mover's distance*. The transportation problem is a special case of the minimum-cost flow problem in complete bipartite graphs. Similarly, the special case of the transportation problem with all supplies and demands equal to 1 is a special case of the minimum-cost perfect matching problem in complete bipartite graphs. For that reason, we refer to this special case as *geometric bipartite matching*.

The Hitchcock-Koopmans transportation problem as defined above is the discrete version of the *optimal transport* or *Monge-Kantarovich* problem originally proposed by Gaspard Monge in 1784 and studied extensively since the early 20th century. Both the continuous and discrete versions of the transportation problem are used in a wide range of applications, including computing the Barycenter of a family of distributions, matching shapes, matching images, and matching probability distributions. We focus on the discrete transportation problem in this section and refer the reader to the book by Villani [75] for more information on (continuous) optimal transport. We begin by discussing algorithms for *geometric bipartite matching*. We then discuss the case of general supplies and demands.

Geometric Bipartite Matching. For simplicity, we slightly reformulate the geometric bipartite matching problem from the version given above. A *matching* M is a set of point-disjoint pairs or *edges* $(r, b) \in R \times B$. A point is *exposed* with respect to M if it does not belong to any pair of M. A *perfect matching* contains every point in $R \cup B$ exactly once. A minimum cost matching is a perfect matching M^* that minimizes $\sum_{(r,b) \in M^*} \mathrm{d}(r, b)$.

In weighted bipartite graphs with n vertices and m edges, a minimum cost matching can be computed in $O(mn + n^2 \log n)$ time using the Hungarian algorithm [47]. The running time has been improved to $O(m\sqrt{n}\,\mathrm{polylog}(n))$ using an interior-point method [57]. This fact, in turn, implies the existence of an $O(n^{5/2}\,\mathrm{polylog}(n))$-time algorithm for geometric bipartite matching.

In the language of geometric bipartite matching, the Hungarian algorithm works as follows. We maintain a set of potentials $\pi : R \cup B \to \mathbb{R}$ on the points of R and B such that $\pi(r) + \pi(b) \leq d(r, b)$ as well as a matching M. Initially, $M = \emptyset$, $\pi(r) = 0$ for all $r \in R$, and $\pi(b) = \min_{r \in R} \mathrm{d}(r, b)$ for all $b \in B$. An edge (r, b) is called *admissible* if $\mathrm{d}(r, b) = \pi(r) + \pi(b)$. The Hungarian algorithm guarantees every edge of M is admissible. An *alternating path* is a sequence that

alternates between edges of M and edges outside of M. An alternating path between two exposed points is called an *augmenting path*.

The algorithm iteratively changes potentials in order to create new augmenting paths lying on admissible edges and then augments M by finding such a path P, removing every edge of $P \cap M$, and adding every edge of $P \oplus M$. Each *augmentation* increases the size of M by one, guaranteeing that each edge of M is admissible. Vaidya [71] showed that each iteration can be performed in $O(n\phi(n))$ time in geometric settings (in contrast to $O(m+n\log n)$ time for general graphs), where $\phi(n)$ is the update time to maintain a weighted bichromatic closest pair between two point sets dynamically under insertions and deletions. For $d = 2$, he described such data structures with $\phi(n) = O(\log^3 n)$ if the distance between a pair of points is measured in the L_1 or L_∞ metric, and with $\phi(n) = O(\sqrt{n})$ for any other L_p metric. Agarwal *et al.* [5] presented an improved data structure for general L_p-metrics with $\phi(n) = O(n^\varepsilon)$, for an arbitrarily small constant $\varepsilon > 0$. The bound was recently improved to $O(\text{polylog}(n))$ by Kaplan *et al.* [55]. Putting these results together, a minimum-weight matching between two point sets in $\mathbb{R}^2$ of size n each can be computed in $O(n^2\,\text{polylog}(n))$ time for any L_p metric. This bound holds for a larger class of well-behaved metrics in the plane; see e.g. [55].

No subquadratic algorithm is known for computing the minimum-weight bipartite matching in the plane except for some very special cases such as when points are in convex position [58] or when points have polynomially bounded integer coordinates [68]. Efrat *et al.* [42] described an $O(n^{3/2}\log n)$ algorithm to compute the optimal bottleneck matching, i.e., minimizing the maximum length of an edge in the matching, in the plane. Consequently, a series of papers have investigated approximation algorithms for geometric bipartite matching [16,73]. Agarwal and Varadarajan [16] proposed a Monte Carlo algorithm that with high probability computes an $O(\log(1/\varepsilon))$-approximate matching in $O(n^{1+\varepsilon})$ time for any fixed dimension and under any L_p-metric. Their algorithm decomposes space using a large randomly shifted grid. Within each grid cell, they recursively compute a matching using as many points as possible. Some points may be left unmatched due to imbalances between the points of R and B lying within each cell, so they optimally match the leftover points lying in all cells using the Hungarian algorithm. In expectation, the total cost of matching the leftover points within each level of recursion is a constant approximation of the cost of the optimal geometric matching on R and B, so limiting the number of levels to $O(\log(1/\varepsilon))$ gives them their desired approximation factor. Adapting their method, Indyk [53] described a Monte Carlo algorithm that in $O(n\,\text{polylog}(n))$ time approximates the cost of the optimal matching within a constant factor, with high probability. Unfortunately, his approach cannot be extended to compute the matching itself.

The state-of-the-art for geometric bipartite matching is an algorithm of Sharathkumar and Agarwal [69] that computes an ε-approximation under any L_p metric in $O(n\,\text{poly}(\log n, 1/\varepsilon))$ time with high probability. Like the Hungarian algorithm, their algorithm performs a sequence of augmentation steps, each

of which increases the size of the matching by one. However, they relax the admissibility condition slightly so that the cost of the matching it computes is at most $(1+\varepsilon)$ times that of the optimal matching but the total number of edges in all the augmenting paths computed is only $O(n\varepsilon^{-1}\log n)$. Using a data structure based on randomly shifted quadtrees, they compute the augmenting path at each step in time proportional to the length of the augmenting path. An interesting question is whether their approach yields an ε-approximation algorithm for the bottleneck matching or the root-mean-square Euclidean matching (i.e., minimize the sum of squares of Euclidean lengths of the matching edges). Agarwal and Sharathkumar [13] presented an algorithm that uses dynamic nearest neighbor data structure and obtains a $O((1/\delta)^{\log_2(1/\delta)})$-approximate matching in $O(n^{1+\delta})$ time. This is the first sub-quadratic $O(1)$-approximation algorithm in high dimensional spaces. For fixed dimensions, they also developed an $O((1/\varepsilon^{O(d)})n^{3/2}\log^5(n/\varepsilon))$-time deterministic algorithm that computes an ε-approximate bipartite matching.

Transportation. We now return to transportation problems with arbitrary integer supplies and demands, with U being the total demand (or supply). As mentioned earlier, the transportation problem can be formulated as an instance of minimum-cost flow in a complete bipartite graph. Using the currently best-known algorithm for minimum cost flow by Lee and Sidford [57], optimal transportation maps can be computed in $O(n^{5/2}\,\mathrm{polylog}(n)\,\mathrm{polylog}(U))$ time, assuming the demand and supply of each point is an integer and the total demand is U. Extending Vaidya's matching algorithm, Atkinson and Vaidya [24] had shown that the optimal transportation map can be computed in $O(n^2\phi(n)\log(U))$ time, where $\phi(n)$ is again the update time for maintaining the weighted bichromatic closest pair. The above discussion implies that the optimal transportation map can be computed in $O(n^2\,\mathrm{polylog}(n)\log(U))$ time under any L_p metric in $\mathbb{R}^2$. Recently, Agarwal *et al.* [10] improved the running time to $O(n^2\,\mathrm{polylog}(n))$ by adapting a strongly polynomial-time minimum-cost flow algorithm by Orlin [64] for general graphs.

Sharathkumar and Agarwal [68] describe an algorithm that computes, in $O((n\sqrt{U}\log^2 n + U\log U)\phi(n)\log(\Delta U/\varepsilon))$ time, where Δ is the diameter of $R\cup B$, a transportation map that is optimal within an additive error of ε. Andoni *et al.* [18] describe a result similar to Indyk's [53] in that they can estimate the *cost* of the optimal transportation map within an ε factor in $n^{1+o_\varepsilon(1)}$ time (here, the constant in the little-o depends upon ε) assuming $U = n^{O(1)}$.

Agarwal *et al.* [10] describe two approximation algorithms for the transportation problem. The first is a Monte Carlo algorithm that computes in $O(n^{1+\varepsilon})$ time a transportation map with expected cost $O(\log^2(1/\varepsilon))$ of the optimal. This algorithm is an extension of the one by Agarwal and Varadarajan [16] for geometric bipartite matching, but a number of new ideas are needed to handle arbitrary demand and supply values. Their other algorithm gives an ε-approximation in $O(n^{3/2}\varepsilon^{-d}\,\mathrm{polylog}(n)\,\mathrm{polylog}(U))$ time. For this algorithm, they construct a graph of $O(n/\varepsilon^d)$ edges over $R\cup B$ for which the optimal transportation distance

is at most $(1+\varepsilon)\mu(\tau^*)$ and then run the minimum cost flow algorithm of Lee and Sidford [57].

4.2 Order Preserving Maps

Let $P = \langle p_1, \ldots, p_n \rangle$ and $Q = \langle q_1, \ldots, q_n \rangle$ be two sequences of points in $\mathbb{R}^d$. Recall, a *correspondence* between P and Q is a set of pairs $C \subseteq P \times Q$. A correspondence C is *monotone* if for any two pairs $(p_i, q_j), (p_{i'}, q_{j'}) \in C$ with $i' \geq i$, we have $j' \geq j$. The goal is to compute monotone correspondences that have certain properties. While one can imagine many criteria for computing monotone correspondences, we focus on three in this survey; minimizing the discrete Fréchet distance, dynamic time warping (DTW), and minimizing the geometric edit distance (GED). These criteria have been used in many applications such as speech recognition, molecular biology, and trajectory analysis.

Both the discrete Fréchet distance and DTW require C to contain every point in P and Q at least once. However, minimizing the Fréchet distance requires finding such a correspondence C that minimizes the maximum distance between p and q for any $(p, q) \in C$ while DTW minimizes the *sum* of distances. Formally, we define the discrete Fréchet distance as $\mathsf{dfr}(P, Q) = \min_C \max_{(p,q)\in C} \|p - q\|$ and the DTW distance as $\mathsf{dtw}(P, Q) = \min_C \sum_{(p,q)\in C} \|p - q\|$ where both minima are taken over all correspondences C covering points in P and Q. GED requires C to be a matching on P and Q—i.e. each point appears in exactly one pair—and it minimizes the sum of distances between matched points plus an additional gap penalty g for each point in P and Q outside of C. Formally, GED is defined as $\mathsf{ged}(P, Q) = \min_C \sum_{(p,q)\in C} \|p - q\| + g(2n - 2|C|)$ with the minimum taken over all matchings C.

Exact Algorithms. All three criteria can be computed in $O(n^2)$ time using straightforward dynamic programming algorithms, and these algorithms can be interpreted as computing a path through a weighted grid G on $V = \{(i, j) \mid 0 \leq i \leq n, 0 \leq j \leq n\}$ where, in general, each cell (i, j) has weight $\|p_i - q_j\|$. The exact paths allowed and their cost vary by criteria, but generally one takes steps from each position (i, j) to one of $(i + 1, j)$, $(i, j + 1)$, and $(i + 1, j + 1)$ while trying to minimize the maximum cell weight encountered for the discrete Fréchet distance or some function based on the sum of weights for DTW or GED. For the remainder of this section, we focus on attempts to improve the running times for computing these criteria both exactly and approximately.

Progress on exact algorithms with worst-case guarantees was only made recently. Agarwal *et al.* [4] described the first result, an $O(n^2 \log\log n / \log n)$ time algorithm for computing the discrete Fréchet distance. We will briefly describe a *decision procedure* for determining if $\mathsf{dfr}(P, Q) \leq \delta$, for a given parameter δ. Then, using a binary search, the actual Fréchet distance can be found. The main idea in the decision procedure is to partition the dynamic programming grid into *blocks* $B_0, B_1, \ldots$ containing $O(\log n)$ columns each. Then, in each block B_i, one needs to (implicitly) determine which of the cells in its rightmost column are *reachable* by a path through the grid touching cells of weight at most δ.

Subsequently, Gold and Sharir [48] described $O(n^2 \log\log\log(n)/\log\log(n))$ time algorithms for computing DTW and GED when $d = 1$. Their approach is similar to [4], but filling the entries of the right column of each B_i is now more challenging, thereby resulting in a somewhat larger running time. For each B_i, they encode the costs of all the paths between boundary cells as points in high dimensional Euclidean space and use a bichromatic dominating pairs reporting algorithm [32] to compute and encode the cheapest boundary-to-boundary paths.

Recent lower bound results suggest that no significantly faster algorithm can be developed for these problems. In particular, no $O(n^{2-\varepsilon})$-time algorithm exists, for constant ε, assuming the strong exponential time hypothesis (SETH) holds[4]. The strongest of these lower bounds was formulated by Abboud *et al.* [1].

Approximations. Bringmann and Mulzer [28] have shown that a strongly subquadratic ε-approximation for the discrete Fréchet distance is unlikely assuming SETH, and they have presented an α-approximation algorithm for this problem, for any $\alpha \in [1, n]$, that runs in $O(n \log n + n^2/\alpha)$ time. In contrast, near-linear-time ε-approximation algorithms have been proposed for discrete Fréchet distance and some other related cost functions by restricting the input to certain *natural families* of sequences. Here we focus on κ-packed curves[5]; see [21] for other types of well-behaved curves. Aronov *et al.* [21] described an ε-approximation for the discrete Fréchet distance that runs in $O((\kappa/\varepsilon)n \log n)$ time on κ-packed sequences (see [9,41]). The key component of their algorithm is an approximate decision procedure that given a value δ, correctly reports, in $O(\kappa n/\varepsilon)$ time, the discrete Fréchet distance to be less than δ if $\mathsf{dfr}(P, Q) < (1 - \varepsilon)\delta$ and correctly reports the distance to be greater than δ if $\mathsf{dfr}(P, Q) > (1 + \varepsilon)\delta$. The decision procedure accomplishes this by constructing simplified sequences P', Q' of P and Q respectively, and then reducing the problem to a reachability problem in a grid graph with $O(\kappa n/\varepsilon)$ cells. Agarwal *et al.* [9] obtained similar bounds for DTW and GED. Their algorithm computes ε-approximations of those criteria in $O((\kappa/\varepsilon)n \log n)$time for κ-packed curves by covering the dynamic programming grid described above with rectangles of similar weights. Then, using the properties of κ-packed curves, they observe that the overall number of boundary cells, appearing along all the rectangles, is relatively small.

4.3 Extensions

Our use of extrinsic cost functions in defining optimal transportation maps and monotone correspondences is useful for matching of shapes that lie in the same position within their ambient spaces. However, one can still use extrinsic costs to do rigid matching between shapes lying in different positions or even orientations in their ambient space by minimizing the measures under translation or rotation.

[4] SETH says that for all $0 < \delta < 1$, there is a $k > 2$ such that k-SAT requires $2^{\delta n}$ time to be solved in the worst-case.

[5] A point sequence is κ-packed if the length of its polygonal chain is at most $\kappa \cdot r$ within any ball of radius r.

One method of finding a good translation/rotation is the iterative closest point (ICP) method [25], wherein one iteratively computes a correspondence between two objects based on an extrinsic measure and then translates/rotates one of the objects to minimize the cost of the correspondence.

Intrinsic costs, mentioned in the beginning of the section, are more useful for applications like non-rigid matching of shapes. Notwithstanding a number of recent results in computer graphics and geometric modeling on methods for intrinsic cost functions—see [54] for a survey—very few combinatorial algorithms are known for them. For example, consider the problem of computing the Gromov-Hausdorff distance between two metric spaces $\mathsf{X}_1 = (X_1, \rho_1)$ and $\mathsf{X}_2 = (X_2, \rho_2)$, defined as

$$\frac{1}{2} \inf_{C} \sup_{(x_1,x_2),(x_1',x_2')\in C} |\rho_1(x_1, x_1') - \rho_2(x_2, x_2')|,$$

where the infimum is taken over all correspondences $C \subseteq X_1 \times X_2$ containing each $x_1 \in X_1$ and $x_2 \in X_2$ at least once. Gromov-Hausdorff distance is used in applications such as matching of deformable shapes [29,60]. The currently known results concerning approximability of computation include a proof that it is NP-Hard to approximate this measure within a factor less than 3, when X_1 and X_2 are metric trees [8,67], and an $O(\min\{n, \sqrt{rn}\})$-approximation algorithm for that case where r is the ratio of the longest edge length in both trees to the shortest edge length [8]. This problem is an instance of low-distortion embedding between two metric spaces, which has been extensively studied; see [59, Chap. 15] for a survey.

5 Discussion

In this survey, we reviewed recent progress on a few geometric optimization problems, namely, geometric set cover and hitting set, geometric independent set, and computing maps between a pair of point sets. Because of lack of space, there are several major developments in geometric optimization that we did not cover here. Perhaps the most significant among them is the PTAS for Euclidean TSP by Arora [22] and by Mitchell [61]. Their techniques—randomly shifted quadtrees and guillotine subdivisions—have been successfully applied to many geometric optimization problems and have had a profound impact on the field. See [23] for a survey.

Another technique that has become quite popular over the last two decades is the *coreset* based approach. Roughly speaking, a coreset is a small subset of the input set where an optimal solution to a problem on the coreset is a good approximation of the optimal solution on the overall set. Surprisingly, fast algorithms exist for computing coresets for a large class of geometric optimization problems whose sizes depend only on the quality of approximation and not on the input size. See surveys [3,66].

Other widely-studied topics include optimal path planning, curve/surface simplification, clustering, and network-design problems.

Acknowledgements. We thank Timothy Chan, Sariel Har-Peled, and Micha Sharir for their helpful comments.

References

1. Abboud, A., Hansen, T.D., Williams, V.V., Williams, R.: Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made. In: Proceedings of the 48th Annual ACM Symposium on Theory Computing, pp. 375–388 (2016)
2. Adamaszek, A., Wiese, A.: Approximation schemes for maximum weight independent set of rectangles. In: Proceedings of the 54th IEEE Annual Symposium on Foundations of Computer Science, pp. 400–409 (2013)
3. Agarwal, P.K., Har-Peled, S., Varadarajan, K.R.: Geometricapproximation via coresets. In: Goodman, J.E., Pach, J., Welzl, E. (eds.) Combinatorial and Computational Geometry, pp. 1–30. Cambridge University Press, New York (2005)
4. Agarwal, P.K., Avraham, R.B., Kaplan, H., Sharir, M.: Computing the discrete fréchet distance in subquadratic time. SIAM J. Comput. **43**, 429–449 (2014)
5. Agarwal, P.K., Efrat, A., Sharir, M.: Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. SIAM J. Comput. **29**(3), 912–953 (1999)
6. Agarwal, P.K., Ezra, E., Ganjugunte, S.K.: Efficient sensor placement for surveillance problems. In: Krishnamachari, B., Suri, S., Heinzelman, W., Mitra, U. (eds.) DCOSS 2009. LNCS, vol. 5516, pp. 301–314. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02085-8_22
7. Agarwal, P.K., Ezra, E., Sharir, M.: Near-linear approximation algorithms for geometric hitting sets. Algorithmica **63**(1–2), 1–25 (2012)
8. Agarwal, P.K., Fox, K., Nath, A., Sidiropoulos, A., Wang, Y.: Computing the Gromov-Hausdorff distance for metric trees. In: Proceedings of 26th International Symposium on Algorithms and Computation, pp. 529–540 (2015)
9. Agarwal, P.K., Fox, K., Pan, J., Ying, R.: Approximating dynamic time warping and edit distance for a pair of point sequences. In: 32nd International Symposium on Computational Geometry, pp. 6:1–6:16 (2016)
10. Agarwal, P.K., Fox, K., Panigrahi, D., Varadarajan, K., Xiao, A.: Efficient algorithms for the geometric transportation problem. In: 33rd International Symposium on Computational Geometry (2017, to appear)
11. Agarwal, P.K., Mustafa, N.H.: Independent set of intersection graphs of convex objects in 2D. Comput. Geom. **34**(2), 83–95 (2006)
12. Agarwal, P.K., Pan, J.: Near-linear algorithms for geometric hitting sets and set covers. In: Proceedings of the 30th Annual Symposium on Computational Geometry, pp. 271–280 (2014)
13. Agarwal, P.K., Sharathkumar, R.: Approximation algorithms for bipartite matchingwith metric and geometric costs. In: Proceedings of the 46th Annual ACM Symposium on Theory of Computing, pp. 555–564 (2014)
14. Agarwal, P.K., Sharir, M.: Algorithmic techniques for geometric optimization. In: van Leeuwen, J. (ed.) Computer Science Today. LNCS, vol. 1000, pp. 234–253. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0015247
15. Agarwal, P.K., Sharir, M.: Efficient algorithms for geometric optimization. ACM Comput. Surv. **30**(4), 412–458 (1998)

16. Agarwal, P.K., Varadarajan, K.R.: A near-linear constant-factor approximation for Euclidean bipartite matching? In: Proceedings of the 20th Annual Symposium on Computational Geometry, pp. 247–252 (2004)
17. Alt, H., Guibas, L.J.: Discrete geometric shapes: matching, interpolation, and approximation. In: Sack, J.R., Urrutia, J. (eds.) Handbook of Computational Geometry, pp. 121 – 153. North-Holland, Amsterdam (2000)
18. Andoni, A., Nikolov, A., Onak, K., Yaroslavtsev, G.: Parallel algorithms for geometric graph problems. In: Proceedings of the 46th ACM Symposium on Theory of Computing, pp. 574–583 (2014)
19. Aronov, B., de Berg, M., Ezra, E., Sharir, M.: Improved bounds for the union of locally fat objects in the plane. SIAM J. Comput. **43**(2), 543–572 (2014)
20. Aronov, B., Ezra, E., Sharir, M.: Small-size ε-nets for axis-parallel rectangles and boxes. SIAM J. Comput. **39**, 3248–3282 (2010)
21. Aronov, B., Har-Peled, S., Knauer, C., Wang, Y., Wenk, C.: Fréchet distance for curves, revisited. In: Azar, Y., Erlebach, T. (eds.) ESA 2006. LNCS, vol. 4168, pp. 52–63. Springer, Heidelberg (2006). https://doi.org/10.1007/11841036_8
22. Arora, S.: Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. J. ACM **45**(5), 753–782 (1998)
23. Arora, S.: Approximation schemes for NP-hard geometric optimization problems: a survey. Math. Program. **97**(1–2), 43–69 (2003)
24. Atkinson, D.S., Vaidya, P.M.: Using geometry to solve the transportation problem in the plane. Algorithmica **13**(5), 442–461 (1995)
25. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**, 239–256 (1992)
26. Bonnet, É., Miltzow, T.: An approximation algorithm for the art gallery problem. CoRR abs/1607.05527 (2016). http://arxiv.org/abs/1607.05527
27. Bonnet, É., Miltzow, T.: Parameterized hardness of art gallery problems. In: 24th Annual European Symposium on Algorithms, vol. 57, pp. 19:1–19:17 (2016)
28. Bringmann, K., Mulzer, W.: Approximability of the discrete fréchet distance. J. Comput. Geom. **7**(2), 46–76 (2016)
29. Bronstein, A.M., Bronstein, M.M., Kimmel, R.: Efficient computation of isometry-invariant distances between surfaces. SIAM J. Sci. Comput. **28**(5), 1812–1836 (2006)
30. Chalermsook, P., Chuzhoy, J.: Maximum independent set of rectangles. In: Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 892–901 (2009)
31. Chan, T.M.: Polynomial-time approximation schemes for packing and piercing fat objects. J. Algorithms **46**(2), 178–189 (2003)
32. Chan, T.M.: All-pairs shortest paths with real weights in $O(n^3/\log n)$ time. Algorithmica **50**(2), 236–243 (2008)
33. Chan, T.M., Grant, E.: Exact algorithms and apx-hardness results for geometric packing and covering problems. Comput. Geom. **47**(2), 112–124 (2014)
34. Chan, T.M., Grant, E., Könemann, J., Sharpe, M.: Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In: Proceedings of the 23rd ACM-SIAM Symposium on Discrete Algorithms, pp. 1576–1585 (2012)
35. Chan, T.M., Har-Peled, S.: Approximation algorithms for maximum independent set of pseudo-disks. Disc. Comput. Geom. **48**, 373–392 (2012)
36. Chekuri, C., Clarkson, K.L., Har-Peled, S.: On the set multicover problem in geometric settings. ACM Trans. Algorithms **9**(1), 9:1–9:17 (2012)
37. Cheong, O., Efrat, A., Har-Peled, S.: Finding a guard that sees most and a shop that sells most. Disc. Comput. Geom. **37**(4), 545–563 (2007)

38. Chuzhoy, J., Ene, A.: On approximating maximum independent set of rectangles. In: Proceedings of the IEEE 57th Annual Symposium on Foundations of Computer Science, pp. 820–829 (2016)
39. Clarkson, K.L., Varadarajan, K.R.: Improved approximation algorithms for geometric set cover. Disc. Comput. Geom. **37**(1), 43–58 (2007)
40. Dinur, I., Steurer, D.: Analytical approach to parallel repetition. In: Proceedings of the 46th Annual ACM Symposium on Theory of Computing, pp. 624–633 (2014)
41. Driemel, A., Har-Peled, S., Wenk, C.: Approximating the Fréchet distance for realistic curves in near linear time. Disc. Comput. Geom. **48**(1), 94–127 (2012)
42. Efrat, A., Itai, A., Katz, M.J.: Geometry helps in bottleneck matching and related problems. Algorithmica **31**(1), 1–28 (2001)
43. Eidenbenz, S., Stamm, C., Widmayer, P.: Inapproximability results for guarding polygons and terrains. Algorithmica **31**(1), 79–113 (2001)
44. Even, G., Rawitz, D., Shahar, S.: Hitting sets when the VC-dimension is small. Inf. Process. Lett. **95**(2), 358–362 (2005)
45. Feige, U.: Approximating maximum clique by removing subgraphs. SIAM J. Discret. Math. **18**(2), 219–225 (2004)
46. Fox, J., Pach, J.: Computing the independence number of intersection graphs. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1161–1165 (2011)
47. Fredman, M.L., Tarjan, R.E.: Fibonacci heaps and their uses in improved network optimization algorithms. J. ACM **34**(3), 596–615 (1987)
48. Gold, O., Sharir, M.: Dynamic time warping and geometric edit distance: Breaking the quadratic barrier. CoRR abs/1607.05994 (2016)
49. Har-Peled, S.: Geometric Approximation Algorithms. American Mathematical Society, Boston (2011)
50. Har-Peled, S.: Quasi-polynomial time approximation scheme for sparse subsets of polygons. In: Proceedings of the 30th Annual Symposium on Computational Geometry, pp. 120–129 (2014)
51. Har-Peled, S., Quanrud, K.: Approximation algorithms for polynomial-expansion and low-density graphs. In: Bansal, N., Finocchi, I. (eds.) ESA 2015. LNCS, vol. 9294, pp. 717–728. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48350-3_60
52. Hochbaum, D.S. (ed.): Approximation Algorithms for NP-Hard Problems. PWS Publishing Co., Boston (1997)
53. Indyk, P.: A near linear time constant factor approximation for Euclidean bichromatic matching (cost). In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 39–42 (2007)
54. van Kaick, O., Zhang, H., Hamarneh, G., Cohen-Or, D.: A survey on shape correspondence. Comput. Graph. Forum **30**(6), 1681–1707 (2011)
55. Kaplan, H., Mulzer, W., Roditty, L., Seiferth, P., Sharir, M.: Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications. In: Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 2495–2504 (2017)
56. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) Complexity of Computer Computations, pp. 85–103. Springer, Boston (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
57. Lee, Y.T., Sidford, A.: Path finding methods for linear programming: solving linear programs in $\tilde{O}$(vrank) iterations and faster algorithms for maximum flow. In: Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science, pp. 424–433 (2014)

58. Marcotte, O., Suri, S.: Fast matching algorithms for points on a polygon. SIAM J. Comput. **20**, 405–422 (1991)
59. Matoušek, J.: Lectures on Discrete Geometry. Springer, New York (2002). https://doi.org/10.1007/978-1-4613-0039-7
60. Mémoli, F., Sapiro, G.: A theoretical and computational framework for isometry invariant recognition of point cloud data. Found. Comput. Math. **5**(3), 313–347 (2005)
61. Mitchell, J.S.B.: Guillotine subdivisions approximate polygonal subdivisions: a simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. SIAM J. Comput. **28**(4), 1298–1309 (1999)
62. Mustafa, N.H., Raman, R., Ray, S.: Settling the APX-hardness status for geometric set cover. In: Proceedings of the 55th IEEE Annual Symposium on Foundations of Computer Science, pp. 541–550 (2014)
63. Mustafa, N.H., Ray, S.: Improved results on geometric hitting set problems. Disc. Comput. Geom. **44**(4), 883–895 (2010)
64. Orlin, J.B.: A faster strongly polynomial minimum cost flow algorithm. Oper. Res. **41**(2), 338–350 (1993)
65. Pach, J., Tardos, G.: Tight lower bounds for the size of epsilon-nets. J. Am. Math. Soc. **26**, 645–658 (2013)
66. Phillips, J.M.: Coresets and sketches. CoRR abs/1601.00617 (2016)
67. Schmiedl, F.: Computational aspects of the Gromov-Hausdorff distance and its application in non-rigid shape matching. Disc. Comput. Geom. **57**(4), 854–880 (2017)
68. Sharathkumar, R., Agarwal, P.K.: Algorithms for the transportation problem in geometric settings. In: Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 306–317 (2012)
69. Sharathkumar, R., Agarwal, P.K.: A near-linear time ε-approximation algorithm for geometric bipartite matching. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing, pp. 385–394 (2012)
70. Urrutia, J.: Art gallery and illumination problems. In: Handbook of Computational Geometry, pp. 973–1027. North-Holland (2000)
71. Vaidya, P.M.: Geometry helps in matching. SIAM J. Comput. **18**(6), 1201–1225 (1989)
72. Varadarajan, K.R.: Weighted geometric set cover via quasi-uniform sampling. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, pp. 641–648 (2010)
73. Varadarajan, K.R., Agarwal, P.K.: Approximation algorithms for bipartite and non-bipartite matching in the plane. In: Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 805–814 (1999)
74. Vazirani, V.V.: Approximation Algorithms. Springer, Heidelberg (2001). https://doi.org/10.1007/978-3-662-04565-7
75. Villani, C.: Optimal Transport: Old and New, vol. 338. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-71050-9
76. Williamson, D.P., Shmoys, D.B.: The Design of Approximation Algorithms. Cambridge University Press, Cambridge (2011)