Fast LP-based Approximations for Geometric Packing and Covering Problems[*]

Chandra Chekuri[†] Sariel Har-Peled[‡] Kent Quanrud[§]

Abstract

We derive fast approximation schemes for LP relaxations of several well-studied geometric optimization problems that include packing, covering, and mixed packing and covering constraints. Previous work in computational geometry concentrated mainly on the rounding stage to prove approximation bounds, assuming that the underlying LPs can be solved efficiently. This work demonstrates that many of those results can be made to run in nearly linear time. In contrast to prior work on this topic our algorithms handle weights and capacities, side constraints, and also apply to mixed packing and covering problems, in a unified fashion. Our framework relies crucially on the properties of a randomized MWU algorithm of [41]; we demonstrate that it is well-suited for range spaces that admit efficient approximate dynamic data structures for emptiness oracles. Our framework cleanly separates the MWU algorithm for solving the LP from the key geometric data structure primitives, and this enables us to handle side constraints in a simple way. Combined with rounding algorithms that can also be implemented efficiently, we obtain the first near-linear constant factor approximation algorithms for several problems.

1. Introduction

Set Cover and Independent Set are two important problems, and are canonical examples of combinatorial covering and packing problems respectively. Both are NP-Hard optimization problems and their approximability is well understood in the worst case [21]. Set Cover can be approximated up to a logarithmic factor, while Independent Set can be approximated only up to quality "close" to n, and almost matching hardness of approximation results are known for these problems. The *geometric* instances of the preceding problems, are usually associated with a range space (P, R), where P is a set of points/elements and $R \subset 2^P$ is a collection of ranges/sets corresponding to some geometric objects such as (pseudo) disks, half-spaces, rectangles, triangles, etc. A range space is a set system that naturally defines an instance of Set Cover or Set Multicover[1]. Independent Set can be viewed as a special variant of packing, where one has to pick maximum number of ranges, such that no element is contained in more chosen ranges than its specified capacity.

Usually, the range space is *implicitly* defined – the geometric entities involved are specified, and the range space arises out of their definition. For example, in the Independent Set problem of a given collection of disks $\mathcal{D}$ in the plane, the locations of the disks are specified. Every face of the arrangement of $\mathcal{A}(\mathcal{D})$ specifies a hyperedge (i.e., set) of disks such that one has to select at most one of them in the independent set.

There has been extensive work on *approximation* algorithms for geometric packing and covering and related problems with several fundamental advances in the last decade [25, 17, 24, 23, 28, 42, 33]. The three main approaches that have been successful are local search, separator based dynamic programming schemes, and LP-relaxation based rounding. These powerful techniques have led to PTASes and QPTASes and constant factor approximations for several problems and these results are in sharp contrast to known strong lower bounds for the general non-geometric settings.

LP rounding in geometry. The LP based approximation algorithms which go back to the influential work of Clarkson [3], whose *reweighting* algorithm can be interpreted as solving and rounding the corresponding LP simultaneously. Clarkson's algorithm was restated for spaces with bounded VC dimension by Brönnimann and Goodrich [5], and the connection to LPs was made explicit by Long [9]. There have been a number of papers in the recent years which have shown the utility of the LP approach for geometric covering and packing problems [13, 26, 25, 18, 24, 23, 37, 40] where they have gone beyond VC dimension to exploit union complexity bounds. The LP approach has been particularly useful for *weighted* and *capacitated* problems where other

[*]This work is supported in part by NSF grants CCF-1526799, CCF-1910149 and CCF-1421231.

[†]Dept. of Computer Science, University of Illinois, Urbana IL 61801. chekuri@illinois.edu. http://chekuri.cs.illinois.edu.

[‡]Dept. of Computer Science, University of Illinois, Urbana IL 61801. sariel@illinois.edu. http://sarielhp.org

[§]Purdue University, West Lafayette, Indiana, 47909. krq@purdue.edu. https://kentquanrud.com.

[1]We do not explicitly consider the Hitting Set problem since it is the same as Set Cover for the dual range space.

approaches are less suitable. The general framework here is to solve an appropriate LP relaxation for the problem followed by a careful rounding procedure that heavily relies on the underlying geometry to improve the approximation bound.

Our goal is to compute fast approximation algorithms for geometric packing and covering problems via the LP approach. The LP relaxations fall under the category of positive linear programs that include pure packing, pure covering and mixed packing and covering. One can write down the LP relaxation explicitly and then apply known techniques for approximating positive linear programs via Lagrangean relaxation and *multiplicative weight update* (MWU) method to obtain several results. There is a vast literature on this topic and we defer a formal discussion to later part of the paper. The running time of these algorithms is near-linear in the number of non-zeroes in the incidence matrix $I(P, R)$ of the range space and can be $O(|P||R|)$ in the worst case. For some problems such as Set Multicover whose relaxation is a mixed covering and packing LP, only bicriteria approximations are feasible via the standard positive linear programming approach.

As we remarked earlier, the range space (P, R) is often supplied in an implicit fashion and the challenge is to obtain a running time that is near-linear in the input size rather than in the size of the incidence matrix $I(P, R)$. There have been significant successes in fast LP solving for implicit instances arising from combinatorial problems. Of particular relevance to this paper is the methodology of combining MWU based methods with data structures which has been successfully used in several settings in graphs and geometry. Before we discuss our contribution we briefly describe some closely related work and some of their limitations.

Limitations of previous work. MWU based methods are used in computational geometry where the reweighting technique of Clarkson and others that we mentioned earlier is routinely used. Most work applies this technique without explicitly referring to LP relaxation, and in fact the technique is very often combined with rounding to directly generate an integer solution. This does not always yield the best known approximation ratio. For instance [22] obtains near-linear approximations for several geometric hitting set problems but the approximation ratios are worse by logarithmic factors in n compared to the best known bounds. Agarwal and Pan [43] developed near linear-time approximation algorithms via the LP approach for unweighted instances of geometric Set Cover and Hitting Set problems (and via duality for some Independent Set problems). They use the high-level paradigm of speeding up MWU based algorithms via geometric data structures. They too, for the most part, round as part of the algorithm. Their work does not address weights or capacities. In [43] the Set Multicover problem is explicitly mentioned as an open problem since their framework did not capture mixed packing and covering constraints. In essence, for geometric packing and covering problems, there has been no systematic effort to efficiently solve the underlying LP relaxations to near-optimality; previous work relied on exact algorithms or known results from positive linear programming in a black-box fashion that did not exploit geometry. In some previous work [36] we demonstrated that for simple and well-behaved range spaces such as intervals and points (and orthogonal boxes in constant dimensions), it is feasible to solve the underlying LP relaxations with weights and capacities efficiently. The main limitation of this work is that the range space had to have an efficient (poly-logarithmic query and update time) dynamic weighted range search data structure which is not available for many range spaces of interest, for instance, disks and points in the plane that we consider here.

Our contributions. We develop a broadly applicable framework to obtain fast approximation schemes for LP relaxations of geometric packing and covering problems. We address limitations of prior approaches and solve some of the problems left open in [43]. We clearly demarcate the boundary between the high-level randomized MWU algorithm from [41] that we build upon, and its efficient implementation via geometric data structures; in particular we abstract the properties we need from the data structure. This clean separation is useful in solving a variety of problems in a unified and modular fashion even when there are side constraints and multiple linear objectives. This will enable future developments in geometric data structures to be easily incorporated into the framework. In order to make our contributions concrete we focus on the canonical setting of disks and points in the plane. We obtain the first nearly linear time approximation schemes for the LP relaxations of the following problems which can be combined with known rounding algorithms.

(A) Maximum weight packing of disks into capacitated points. Given a collection of weighted disks and capacitated points, compute a subcollection of disks of maximum weight such that the number of disks containing any point is at most its capacity. Ene *et al* [37] presented a rounding scheme for that yields constant factor approximation.

(B) Maximum weight independent set of disks. Given a collection of weighted disks in the plane, compute a subcollection of disjoint disks of maximum weight. Chan and Har-Peled [25] showed that the fractional solution to the underlying LP can be rounded to give a constant factor approximation to this problem. This problem can be posed as a discrete packing problem (as above) with a quadratic number of points in the plane, but we obtain running times nearly linear in the number of disks.

(C) Weighted set cover of points by disks. Given a collection of weighted disks in the plane and a set of points, the problem is to compute a subcollection of disks of minimum weight, such that each point is covered by at least one disk. Chan *et al* [24] (extending the work by Varadarajan [18]) showed how to round the LP to get a constant factor approximation to the discrete problem.

(D) Minimum weight multi-cover of points by disks. Given a collection of weighted disks in the plane and capacitated points, the problem is to compute a subcollection of disks of minimum weight such that the number of disks lying above any point is at least the capacity of that point. [26] showed that the fractional solution to the underlying LP can be rounded to give a constant factor approximation to the unweighted problem and Bansal and Pruhs [23] extended the work of [24] to obtain a constant factor for the weighted case. We obtain a bicriteria approximation scheme for the underlying LP relaxation. When the demands are not too large the bicriteria approximation can be converted into unicriteria approximation. Using knapsack-cover inequalities and an algorithm of [44] we also describe an $O(mn)$-time approximation scheme for the LP. As far as we are aware no fast approximation scheme for the LP was known previously and [43] posed it as an open problem.

Our framework applies to any range space that admits efficient emptiness oracles with deletions. We give a formal description and example later in the paper.

Side constraints and mixed packing and covering: We highlight a particularly useful aspect of our framework via two examples. Consider the problem of covering points by weighted disks. We typically have a single linear objective function but there are several applications where there are multiple linear cost functions that we wish to simultaneously minimize (usually a small number). In such cases we need to solve the LP relaxation for the problem with multiple linear costs which are formulated via budget constraints. This results in a mixed packing and covering LP where the packing constraints modeling the upper bounds on the costs do not have much structure while the covering constraints come from the geometric setting. Similarly, when considering maximum independent set of disks, we may have multiple objectives that we wish to simultaneously maximize which results again in a mixed packing and covering LP. We illustrate a different motivation for side constraints. Consider again the problem of maximum independent set of disks. It is natural in several applications to have additional packing constraints. For example there could be partition matroid constraints on choosing disks: in more friendly language this corresponds to the setting where disks are partitioned into color classes and there is an prescribed upper bound on the number of disks that can chosen from each color class. Rounding LP relaxations to handle multiple types of packing constraints is often feasible via the approach of contention resolution schemes [27]. Our framework allows us to cleanly handle the implicit constraints imposed by the geometric range space and the explicit constraints that may not have much additional structure.

Rounding the LP relaxation: For many problems of interest the procedure for rounding the solution to the LP can be done in near linear time with some basic preprocessing and data structures — for example see [25, 43] for rounding the setting of independent set of disks. Here, we mainly focus on solving the LP relaxation. Full details of the rounding schemes are beyond the scope of this current version.

2. Background and formal statement of results

We state our results using the following abstraction.

Definition 2.1. *A* range space *is pair (P, R), where P is a set of* points *(or elements), and $R \subseteq 2^P$ is a collection of subset of P called ranges. We let $m = |P|$ denote the number of points (when P is finite) and $n = |R|$ denote the number of ranges.*

We make the standard technical assumption that a set of ranges R is always in "general position"; in particular, any two ranges in R intersect at most a finite number of times, and any intersection is a proper crossing.

Definition 2.2. *A* positive linear program *(a.k.a. a mixed packing and covering LP) is a linear program of the form*

$$\text{find } x \in \mathbb{R}^n_{\geq 0} \text{ s.t. } Ax \leq \mathbb{1} \text{ and } Bx \geq \mathbb{1}, \tag{2.1}$$

where $A \in \mathbb{R}^{m_p \times n}_{\geq 0}$ and $B \in \mathbb{R}^{m_c \times n}_{\geq 0}$ have nonnegative coefficients.

Note that there is no objective function in Eq. (2.1). One can model an objective with positive coefficients as a constraint; since both packing and covering constraints are allowed, both maximization and minimization objectives can be modeled.

Several applications fall into the setting of pure packing where we aim to solve an LP of the following form where $c \in \mathbb{R}_{\geq 0}$,

$$\max\langle c, x\rangle \text{ s.t. } Ax \leq \mathbb{1} \text{ and } x \geq \mathbb{0} \tag{2.2}$$

and pure covering where we aim to solve an LP of the following form,

$$\min\langle c, x\rangle \text{ s.t. } Bx \geq \mathbb{1} \text{ and } x \geq \mathbb{0}. \tag{2.3}$$

Definition 2.3. *A vector $y \in \mathbb{R}^n_{\geq 0}$ is a $(1 \pm \varepsilon)$-*approximate *solution to a positive linear program if it satisfies the property that $Ay \leq (1+\varepsilon)\mathbb{1}$ and $By \geq (1-\varepsilon)\mathbb{1}$.*

We note that if y is a $(1 \pm \varepsilon)$-approximation solution one can, by scaling, also obtain a solution that satisfies the packing constraints and only violates the covering constraints by a $(1 - O(\varepsilon))$-factor. Similarly one can obtain a solution that satisfies the covering constraints and only violates the packing constraints by a $(1+O(\varepsilon))$-factor.

Definition 2.4. *Given a range space (P, R), let $I(P, R)$ (or simply I, when (P, R) is clear) be the incidence matrix between the points and ranges. Formally, we have $I \in \{0, 1\}^{m \times n}$, with $I(p, r) = 1$ if $p \in r$ and $I(p, r) = 0$ otherwise, where $m = |P|$ and $n = |R|$.*

Geometric packing and covering problems are induced by the incident matrix I. For instance, we get the following LPs:

1. Weighted Set Cover:
$$\min \sum_i c_i x_i \text{ s.t. } Ix \geq \mathbb{1} \text{ and } x \geq \mathbb{0}.$$
2. Weighted Set Multicover:
$$\min \sum_i c_i x_i \text{ s.t. } Ix \geq d \text{ and } \mathbb{0} \leq x \leq \mathbb{1}.$$
3. Packing weighted ranges into points:
$$\max \sum_i c_i x_i \text{ s.t. } Ix \leq \mathbb{1} \text{ and } x \geq \mathbb{0}.$$

In some cases, points and ranges can have non-negative capacities and demands associated with them. We define a general class of positive LPs induced by a range space.

Definition 2.5. *Let (P, R) be a range space. A matrix $A \in \mathbb{R}^{m \times n}$* is associated *with (P, R) if it can be written as $A = \operatorname{diag}(\alpha) I \operatorname{diag}(\beta)$, where $\alpha \in \mathbb{R}^m_{\geq 0}$, $\beta \in \mathbb{R}^n_{\geq 0}$, and $I = I(P, R)$.*

A matrix associated with (P, R) can be interpreted as assigning weights to the points (i.e., α) and to the ranges (i.e., β). This work is concerned with solving positive linear programs with packing and covering constraints induced by a geometric range system (P, R), without explicitly computing the incidence matrices. To solve such implicitly defined LP's in time faster than the explicit size of the LP, we require a data structure for the range system (P, R), which has been established for several geometric range spaces.

Definition 2.6. *Let (P, R) be a fixed range system. An* emptiness oracle *is a (randomized) data structure that takes as initial input a subset $Q \subseteq P$, and given a query range $r \in R$ returns whether $Q \cap r = \emptyset$ or not. An* emptiness oracle with deletions *is an oracle that also allows deletions from the set Q.*

Remark 2.1. *Randomization is crucial to obtain efficient emptiness oracles. In our application we will assume that the randomized data structures are either Las Vegas or that they answer* all *queries (which may be adaptive) correctly with high probability.*

Related queries include *count queries*, that return $|Q \cap r|$, and *reporting queries*, that list the elements of $Q \cap r$. When the range system (P, R) takes on different geometries, one can obtain different tradeoffs in performance for these queries. We abstract out specific geometric considerations by assuming access to an "efficient" emptiness oracle, where our definition of efficiency is somewhat loose.

Suppressing logarithmic factors: There is a deep and extensive literature determining tight sublinear factors in geometric settings. However, it is difficult to account for the varying logarithmic factors of each setting precisely without breaking our analysis into very detailed cases. Moreover, the basic run-time improvements in this work are by larger orders of magnitude, from quadratic or more to nearly-linear. At this level of granularity, we do not emphasize logarithmic factors, and for ease of exposition, we adopt the following convention.

Notation 2.1. *The notation $\widetilde{O}(\cdot)$ hides polynomial factors in $\log(m)$, $\log(n)$, and $\log(1/\varepsilon)$. Moreover, a event holds "with high-probability" if the event occurs with probability at least $1 - 1/\operatorname{poly}(m, n)$.*

Definition 2.7. *A range space has* **nearly linear emptiness oracle** *if an emptiness oracle can be implemented with (expected) initialization time $\widetilde{O}(|Q|)$ for*

$Q \subseteq P$ and (expected) query time $\widetilde{O}(1)$. A range space has nearly linear emptiness oracle with deletions *if it supports deletions from Q in $\widetilde{O}(1)$ amortized time.*

We list below some important classes of range spaces for which we have nearly linear emptiness oracles with deletion:

- (P, R) when P is a set of points and R is a set of half-spaces in $\mathbb{R}^3$ [16, 38]. This also holds for the dual range space. Via standard reduction, this also holds when P is a set of points and R is a set of disks in the plane and its dual range space.
- P is a set of points in the plane and R is a set of fat triangles [20]. Also for the dual range space where P is a set of fat triangles and R is a set of points [8].
- P is a set of points and R is a set of orthogonal boxes in $\mathbb{R}^d$ where d is a fixed constant (also for the dual range space). For these shapes there are efficient weighted range search data structures.

Improved approximations for geometric range spaces such as (pseudo) disks and other shapes have been shown via the notion of union complexity which plays a crucial role in both running times and approximation bounds.

Definition 2.8. *A set R of regions in the plane has* low-union complexity, *if*
(A) every range in R has finite descriptive complexity,
(B) the boundaries of any pair of ranges of R intersect only a constant number of times,
(C) and, for any subset $X \subseteq R$ of t ranges, the descriptive complexity of the boundary of the union of $\bigcup X$, is bounded by $O(tf(t))$, where $f(t) = o(t)$ (usually $f(t) = O(\operatorname{poly}\log t)$).

Some ranges for which near linear union complexity is known are the following. We refer the reader to a comprehensive survey [14] for the extensive literature on this topic.

- Pseudo disks (and hence disks) in the plane.
- Fat triangles in the plane and generalizations to (α, β)-covered objects [11].
- Half spaces and axis aligned unit cubes in $\mathbb{R}^3$.

Remark 2.2. *In this paper we focus on range spaces with nearly linear emptiness oracles. Our framework, however, applies to range spaces with less efficient oracles as well. For instance if a range space admits an emptiness oracle with $O(\sqrt{m})$ query time we can obtain a running time of the form $\widetilde{O}((m+n)\sqrt{m})$ to solve the associated LPs. A more detailed treatment will be done in a future version.*

We assume that all the nonzero entries in the input are within a $\operatorname{poly}(m, n, 1/\varepsilon)$-multiplicative factor of each other. When working with $(1+\varepsilon)$-approximation and relaxations of $\{0,1\}$ positive integer programs we can arrange this by simple scaling ideas.

2.1. Results

We now describe our results. The first theorem handles positive LP s that consist of packing constraints induced by a range space with nearly linear emptiness oracle, along with mixed packing and covering constraints that are given explicitly.

Theorem 2.1. *Let (P, R) be a range system with nearly linear emptiness oracle, a parameter $\varepsilon > 0$, $m = |P|$, and $n = |R|$. Consider a positive linear program of the form*

$$\text{find } x \in \mathbb{R}^n_{\geq 0} \text{ s.t. } Ix \leq \mathbb{1},\ Ax \leq \mathbb{1}, \text{and } Bx \geq \mathbb{1},$$

where $I \in \mathbb{R}^{m \times n}_{\geq 0}$ is associated with (P, R), and A and B are matrices. One can compute, with high probability, in $\widetilde{O}\big(N\varepsilon^{-1} + \operatorname{rows}(B)\varepsilon^{-2} + (m+n)\varepsilon^{-4}\big)$ time, an $(1 \pm \varepsilon)$-approximation to the LP (see Definition 2.3), where $N = \|A\|_0 + \|B\|_0$.

The next theorem extends the previous theorem to allow both packing and covering constraints associated with (P, R) when (P, R) has stronger, nearly linear emptiness oracle with deletions.

Theorem 2.2. *Let (P, R) be a range system with nearly linear emptiness oracle with* deletions, *and $\varepsilon > 0$. Consider a positive linear program of the form*

$$\begin{aligned} &\text{find } x \in \mathbb{R}^n_{\geq 0} \\ &\text{s.t. } Ax \leq \mathbb{1},\ B'x \leq \mathbb{1},\ Bx \geq \mathbb{1}, \text{ and } C'x \geq \mathbb{1}, \end{aligned}$$

where $A, B \in \mathbb{R}^{m \times n}_{\geq 0}$ are associated with (P, R). Then, one can $(1 \pm \varepsilon)$-approximate this LP, with high probability, in $\widetilde{O}\big(N\varepsilon^{-1} + \operatorname{rows}(C')\varepsilon^{-2} + (m+n)\varepsilon^{-4}\big)$ randomized time, where $N = \|B'\|_0 + \|C'\|_0$.

The preceding theorems allow us to efficiently solve a variety of geometric packing, covering, and mixed packing and covering LPs induced by geometric range spaces. Note that pure covering and pure packing problems can be reduced to mixed packing and covering by guessing the optimal objective value (using binary search) and encoding the objective as a packing or covering constraint. One can also treat them separately, which in some cases leads to more intuitive algorithms. For the sake of concreteness, we list a few applications for disks and points in the plane.

COROLLARY 2.1. *Given m capacitated points and n weighted disks, one can $(1 \pm \varepsilon)$-approximate, with high probability, the maximum weight* fractional *packing of the disks into the capacitated points, in $\widetilde{O}\big((m+n)\varepsilon^{-4}\big)$ time.*

COROLLARY 2.2. *Given m points in the plane and n weighted disks, one can $(1 \pm \varepsilon)$-approximate, with high probability, the maximum weight* fractional *set cover of the points by the disks in $\widetilde{O}\big((m+n)\varepsilon^{-4}\big)$ time.*

For mixed packing and covering, we obtain a bicriteria approximation guarantee, as seen in the following. The bicriteria approximation can be avoided by incorporating knapsack covering constraints and using the algorithm from [44] which we state subsequently.

DEFINITION 2.9. *An instance of Set Multicover is a tuple (P, R, d, c), where $d : P \to \mathbb{Z}_{\geq 0}$ are the demands on the points, and $w : R \to \mathbb{R}_{>0}$ are the weights on the ranges. A feasible fractional solution assigns a fraction to each range $x : R \to [0, 1]$, such that for any $p \in P$, we have that $\sum_{r \in R(p)} x_r \geq d_p$, where $R(p) = \{r \in R \mid p \in r\}$. An integer solution satisfies the additional property that $x \in \{0, 1\}^R$.*

COROLLARY 2.3. *Given an instance of Set Multicover (P, R, d, w), where the ranges R are disks in the plane, and a parameter $\varepsilon > 0$. Then, one can compute, with high probability, a fractional feasible solution x for $(P, R, (1-\varepsilon)d, c)$ with $c(x) \leq \mathrm{OPT}$ where OPT is the value of an optimum fractional solution. The running time of the algorithm is $\widetilde{O}\big((m+n)\varepsilon^{-4}\big)$. This also implies that a fractional feasible solution x for the original instance with $c(x) \leq (1+\varepsilon)\,\mathrm{OPT}$ can be computed with high probability in time $\widetilde{O}\big((m+n)d_{\max}^4\varepsilon^{-4}\big)$ where $d_{\max}$ is the maximum demand.*

The preceding corollary allows us to obtain a fast algorithm for Set Multicover if one settles for a bicriteria approximation or when the maximum demand is small. If one wants a unicriteria approximation, fast approximation schemes for positive linear programs are not directly helpful other than choosing ε to be very small. However, we show that one can use knapsack cover inequalities and a result from [44] to obtain the following.

COROLLARY 2.4. *Given an instance of Set Multicover (P, R, d, w), where the ranges R are disks in the plane, and a parameter $\varepsilon > 0$. Then, one can compute, with high probability, a fractional feasible solution x for (P, R, d, c) with $c(x) \leq (1+\varepsilon)\,\mathrm{OPT}$ where OPT is the value of an optimum fractional solution. The running time of the algorithm is $\widetilde{O}\big(N/\varepsilon^{-3} + (m+n)/\varepsilon^5\big)$ where N is the number of nonzeroes in $I(P, R)$.*

Theorem 2.1 and Theorem 2.2 obtain nearly linear running time with respect to $|P| + |R|$, i.e., when the range space (P, R) is given explicitly. The next theorem allows us to consider a set of problems where the range space is continuous or implicitly defines a range space (P, R) where $|P| + |R|$ is larger than the input size.

THEOREM 2.3. *Let R be a weighted set of n shapes in the plane, with low-union complexity, that can be maintained efficiently under deletions and emptiness stabbing queries (i.e., each operation can be done in amortized polylogarithmic time). Then one can compute, in $\widetilde{O}\big(n/\varepsilon^4\big)$ time, with high probability., a $(1 \pm \varepsilon)$-approximation to the fractional solution of the following LP (for the maximum weight independent set):*

$$\max_{x \geq 0} \sum_{r \in R} c_r x_r \ \ s.t. \ \ \sum_{r \in R(p)} x_r \leq 1 \quad \forall p \in \mathbb{R}^2,$$

where c_r is the weight of r.

Combining the preceding theorem with the rounding in [25] we obtain the following corollary.

COROLLARY 2.5. *Given a collection of n weighted disks in the plane and $\varepsilon > 0$, one can compute an $(1-\varepsilon)$-approximation to the maximum weight* fractional *independent set in near linear time. By rounding it, one can compute a constant factor approximation to the maximum weight discrete independent set in near linear time. The algorithm succeeds with high probability.*

REMARK 2.3. *Consider the problem of minimum weight dominating set in a collection of disks in the plane. One can obtain a constant factor approximation via the natural LP relaxation [39] even in the more general setting of pseduo disks. For disks we can obtain a near-linear time approximation scheme to solve the LP via Theorem 2.2 and a geometric transformation that converts the dominating set problem into a problem about covering points by cones in three dimensions, and using certain facts about their arrangement [38]. We defer details of this reduction to a future version of the paper.*

Efficient emptiness oracles are much more easy to develop than emptiness oracles that handle deletions. We need deletions to handle covering constraints that get dropped after they are sufficiently covered and this is crucial for width-independent running times. It is an interesting open problem whether the MWU framework can be revisited from a more general perspective so that that can be avoided.

2.2. Technical ideas and other related work Our work is inspired by two streams of work. The first is the already mentioned line of work which showed that LP relaxations can be used to obtain improved approximation algorithms for a certain class of geometric packing and covering problems. The LP approach is particularly useful in the weighted setting as well as in capacitated settings. The second line of work is on approximation schemes for positive LPs starting with work of Grigoriadis and Khachiyan [4] and Plotkin, Shmoys and Tardos [6] and many subsequent developments. This line of work has led to the so-called *width-independent* running times for positive LPs which culminated in a deterministic algorithm for an explicitly given mixed packing and covering LP whose running time is $O(\frac{1}{\varepsilon^2} N \log N)$ where N is the number of non-zeroes in the input [30]. There have been several recent exciting improvements that have been inspired by techniques from continuous optimization and this has resulted in algorithms with a better dependence on ε for pure packing [31] and pure covering [34].

LPs that arise in discrete and combinatorial optimization problems such as graphs, geometry and other areas have additional structure and are often implicit (in some cases the size of the LP is exponential in the input size). For such problems the best running time is obtained by combining several ideas. In this paper we rely on MWU type algorithms and here the properties of MWU algorithm, problem-specific data structures, and their *interplay* is crucial. Recent work [36, 35, 41] has adapted ideas from [30, 29] to develop fast algorithms for a variety of LPs that arise in combinatorial optimization. [36] obtained near-linear deterministic algorithms for some geometric packing problems which involved axis aligned boxes in low dimensions by taking advantage of the structure of the range search data structures. One of the algorithms of Agarwal and Pan [43] is quite similar to that of Koufagiannis and Young [29] which is based on a randomized two player game.

The limitation of the algorithm of [43, 29] is two fold. First, it applies only to pure packing and covering LPs. Second, the primal-dual algorithm requires one to explicitly maintain both primal and dual variables and in some implicit settings this is not suitable. To overcome the preceding limitations we rely on the randomized MWU algorithm in [41] which is inspired by [29, 30] — some features of this algorithm were partly motivated by geometric packing and covering problems involving disks and points in the plane which were not amenable to the techniques in [36]. In a sense [41] combines the correlated weight update feature of [29] with ideas from [30] to obtain a randomized width independent algorithm for mixed packing and covering. The correlated weight update is a key ingredient that we exploit here. In addition the algorithm does not need to explicitly maintain the primal variables as long as certain oracle is available. This is particularly useful in implicit settings as we will demonstrate.

We set up a general framework that essentially abstracts away all the complexity of weights, capacities, and side constraints to the availability of efficient emptiness oracles (with deletion) for the underlying range space. A key technical challenges is to implement the weight decrease in the MWU framework for covering constraints with only access to an emptiness oracle with deletions.

3. Randomized MWU for Positive LPs

In this section we give an overview of a randomized MWU algorithm from [41] that forms the basis for the results in the paper. We set up some useful notation.

NOTATION 3.1. *It would be convenient to consider the coordinates of a vector as indexed by the objects under consideration. As such, for a set of objects $\mathcal{C}$, we use $\mathbb{R}^{\mathcal{C}}$ to denote the space $\mathbb{R}^m$, where $m = |\mathcal{C}|$. Thus, for a vector $v \in \mathbb{R}^{\mathcal{C}}$, and an object $o \in \mathcal{C}$, we denote by v_o the coordinate of v that corresponds to o. In particular, for a set $\mathcal{Q} \subseteq \mathcal{C}$, and vectors $x, y \in \mathbb{R}^{\mathcal{C}}$, let $\langle x, y \rangle_{\mathcal{Q}} = \sum_{i \in \mathcal{Q}} x_i y_i$.*

Consider a positive LP

$$\text{find } x \geq \mathbb{0} \text{ s.t. } Ax \leq \mathbb{1}, Bx \geq \mathbb{1} \tag{3.4}$$

The first component appropriated from [41][2], called `random-mwu` and sketched in Figure 1, is (at a high-level) a variant of a deterministic algorithm for mixed packing and covering originally proposed in [10] and refined in [30]. The algorithm falls in the broad framework of Lagrangian relaxation algorithms that iteratively solve a relaxation of the original problem as follows. The algorithm maintains non-negative weights for each constraint (which can be interpreted as dual variables). We let v denote the weight vector for packing constraints $Ax \leq \mathbb{1}$, and w denote the vector for covering constraints $Bx \geq \mathbb{1}$; v and w are both initialized to the all-1's vector $\mathbb{1}$. In each iteration, the algorithm uses v and w to collapse the packing and covering constraints in to a single packing and covering and finds a feasible solution y to the following relaxed problem:

$$\text{find } x \geq \mathbb{0} \text{ s.t. } \langle v, Ax \rangle \leq \langle v, \mathbb{1} \rangle,\ \langle w, Bx \rangle \geq \langle w, \mathbb{1} \rangle \tag{3.5}$$

[2]We note that for pure packing and pure covering problems, one can also instrument the framework of [29] to obtain some similar results.

```
random-mwu($A \in \mathbb{R}_{\geq 0}^{\mathcal{P}\times n}$,$B \in \mathbb{R}_{\geq 0}^{\mathcal{C}\times n}$,$\varepsilon$)
1. $\eta = (\ln m)/\varepsilon$          // parameter to control step size
2. $v \leftarrow \mathbb{1}$, $w \leftarrow \mathbb{1}$          // packing & covering weights
3. $\mathcal{Q} \leftarrow [\mathcal{C}]$          // $\mathcal{Q}$: active covering constraints
4. $t \leftarrow 0$          // time goes from 0 to 1
5. while $t \leq 1$ and $\mathcal{Q} \neq \emptyset$
   A. choose $y \in \mathbb{R}_{\geq 0}^n$ such that
      /* y is an approximate solution to Lagrangian
         relaxation w/r/t weights $v, w$          */
      1. $\langle v, Ay\rangle \leq (1+O(\varepsilon))\langle v, \mathbb{1}\rangle$
      2. $\langle w, By\rangle_{\mathcal{Q}} \geq (1-O(\varepsilon))\langle w, \mathbb{1}\rangle_{\mathcal{Q}}$
   B. if no $y \in \mathbb{R}_{\geq 0}^n$ satisfies (*) and (**) then
      return "infeasible"
   C. $\delta \leftarrow$ max value $\delta > 0$ such that    // step size
      • $\delta\eta Ay \leq \varepsilon\mathbb{1}$
      • $\delta\eta By \leq \varepsilon\mathbb{1}$
      • $t + \delta \leq 1$
   D. $x \leftarrow x + \delta y$ // increment current solution by $\delta y$
   E. $t \leftarrow t + \delta$          // increment time
   F. pick $\theta \in [0,1]$ uniformly at random
   G. for $i \in \mathcal{P}$          // update packing weights
      // approximate $v_i \leftarrow \exp(\delta\eta\langle e_i, Ay\rangle)v_i$
      1. if $\theta \leq \delta\eta\langle e_i, Ay\rangle/\varepsilon$
         a. $v_i \leftarrow \exp(\varepsilon)v_i$
   H. for $i \in \mathcal{Q}$     // update active covering weights
      // approximate $w_i \leftarrow \exp(-\delta\eta\langle e_i, By\rangle)$
      1. if $\theta \leq \delta\eta\langle e_i, By\rangle/\varepsilon$
         a. $w_i \leftarrow \exp(-\varepsilon)w_i$
         b. if $w_i \leq \exp(-\eta)$
            // i made inactive if weight small enough
            1. $\mathcal{Q} \leftarrow \mathcal{Q} - i$
6. return $x$
```

Figure 1: *A randomized, width-independent implementation of the MWU framework from [41].*

The relaxed problem is infeasible only if the original problem is infeasible. The basic observation motivating this approach is the relative simplicity of (3.5) compared to (3.4). For $1 \leq i \leq n$, let $\alpha_i = \langle v, A\rangle_i / \langle v, \mathbb{1}\rangle$ and let $\beta_i = \langle w, B\rangle_i / \langle w, \mathbb{1}\rangle$. Then (3.5) is feasible iff there exists an i such that $\alpha_i/\beta_i \leq 1$. Setting $y_i = 1/\beta_i$ and all other coordinates to zero is a feasible solution. The fact that y is *supported by a single coordinate* is instrumental to fast running times. Moreover, random-mwu requires only approximate solutions to the relaxed problem. A $(1+\varepsilon)$-approximate solution to the relaxed problem translates to a $(1\pm\varepsilon)$-approximate solution to the original problem. Approximation offers considerable flexibility and leads to substantial improvements in the running time.

The algorithm adds the solution y to the current solution (which is initialized to $\mathbb{0}$) with an appropriate step size δ. The algorithm follows the "timed" framework from [32] that indexes progress by a "time" t that increases from 0 to 1, with the step size and other parameters appropriately normalized. After each iteration the packing and covering weights are updated multiplicatively (packing weights are increased in an exponential fashion and covering weights are decreased).

The efficiency of the algorithm depends on the number of iterations and the work done in each iteration. The randomized algorithm is shown to terminate with high probability in $O(m \log m/\varepsilon^2)$ iterations where m is the total number of constraints in the LP. Each iteration requires two main steps: (i) finding a solution to (3.5), and (ii) updating the weights. The key to implementation efficiency is the randomized weight update step that is borrowed from the work of [29]. Where the standard deterministic update might increase a weight by a multiplicative factor of $\exp(\varepsilon p)$ for some $p \in [0,1]$, random-mwu increases the weight by a multiplicative factor of $\exp(\varepsilon)$ with probability p. In expectation, random-mwu makes the appropriate update with respect to the logarithm of the weight. The crucial property is that all the weight updates are *correlated* via a single random variable θ. The pseudocode in Figure 1 is incomplete, as we leave the implementation of lines (5.A) and (5.C) unspecified. One can take advantage of this in implicit problems, where (5.A) and (5.C) can be supplied by domain-specific oracles. The correlated weight update steps (5.G.i) and (5.H.i) can also be implemented efficiently in implicit settings as we will discuss shortly.

random-mwu-pack (see Figure 2) and random-mwu-cover (see Figure 3) specialize random-mwu to pure packing and pure covering problems. [41] proved that random-mwu terminates both successfully and efficiently with high probability.

THEOREM 3.1. ([41]) *Let $A \in \mathbb{R}_{\geq 0}^{m_p \times n}$ and $B \in \mathbb{R}_{\geq 0}^{m_c \times n}$ be nonnegative matrices for which there exists a nonnegative $x \in \mathbb{R}_{\geq 0}^n$ such that $Ax \leq \mathbb{1}$ and $Bx \geq \mathbb{1}$. Let $m = m_p + m_c$, and let N be the total number of nonzero coefficients in A and B.*

With probability $1 - 1/\operatorname{poly}(m)$, random-mwu($A$,$B$,$\varepsilon$) *returns a point $\hat{x}$ such that $A\hat{x} \leq (1+O(\varepsilon))\mathbb{1}$ and $B\hat{x} \geq (1-O(\varepsilon))\mathbb{1}$ in $O\big((m_c + \min\{m_p, n\})\ln(n)/\varepsilon^2\big)$ iterations and, excluding the time spent in lines* (5.A) *and* (5.C)*,*

```
random-mwu-pack($A \in \mathbb{R}_{\geq 0}^{\mathcal{P} \times n}$, $c \in \mathbb{R}_{\geq 0}^n$, $\varepsilon$)
1. $\eta = (\ln m)/\varepsilon$          // parameter to control step size
2. $v \leftarrow \mathbb{1}$                        // v: packing weights
3. $t \leftarrow 0$                        // time goes from 0 to 1
4. while $t \leq 1$
   A. choose $y \in \mathbb{R}_{\geq 0}^n$ such that
        $\langle c, y\rangle \geq (1 - O(\varepsilon)) \max\{\langle c, z\rangle : \langle v, Az\rangle \leq \langle v, \mathbb{1}\rangle\}$
   // y is APX soln to Lagrangean relaxation w/r/t v
   B. $\delta \leftarrow$ max value $\delta > 0$ such that   // step size
      1. $\delta\eta Ay \leq \varepsilon\mathbb{1}$
      2. $t + \delta \leq 1$
   C. $x \leftarrow x + \delta y$ // increment current solution by $\delta y$
   D. $t \leftarrow t + \delta$                      // increment time
   E. pick $\theta \in [0, 1]$ uniformly at random
   F. for $i \in \mathcal{P}$    // update/increase packing weights
      1. if $(\theta \leq \delta\eta\langle e_i, Ay\rangle/\varepsilon)$
         // approximate $v_i \leftarrow \exp(\delta\eta\langle e_i, Ay\rangle)v_i$
         a. $v_i \leftarrow \exp(\varepsilon)v_i$
5. return $x$
```

Figure 2: *A randomized, width-independent implementation of the MWU framework from [41] specialized to pure packing problem of the form* $\max\langle c, x\rangle$ *s.t* $Ax \leq \mathbb{1}, x \geq \mathbb{0}$.

```
random-mwu-cover($B \in \mathbb{R}_{\geq 0}^{\mathcal{C} \times n}$, $c \in \mathbb{R}_{\geq 0}^n$, $\varepsilon$)
1. $\eta = (\ln m)/\varepsilon$          // parameter to control step size
2. $w \leftarrow \mathbb{1}$                           // w: covering weights
3. $\mathcal{Q} \leftarrow [\mathcal{C}]$            // Q: active covering constraints
4. $t \leftarrow 0$                           // time goes from 0 to 1
5. while $t \leq 1$ and $\mathcal{Q} \neq \emptyset$
   A. choose $y \in \mathbb{R}_{\geq 0}^n$ such that
        $\langle c, y\rangle \leq (1 + O(\varepsilon)) \min\{\langle c, z\rangle : \langle w, Bz\rangle_{\mathcal{Q}} \geq \langle w, \mathbb{1}\rangle_{\mathcal{Q}}\}$
   // y is APX soln to Lagrangean relaxation w/r/t w
   B. $\delta \leftarrow$ max value $\delta > 0$ such that   // step size
      1. $\delta\eta By \leq \varepsilon\mathbb{1}$
      2. $t + \delta \leq 1$
   C. $x \leftarrow x + \delta y$ // increment current solution by $\delta y$
   D. $t \leftarrow t + \delta$                      // increment time
   E. pick $\theta \in [0, 1]$ uniformly at random
   F. for $i \in \mathcal{Q}$    // decrease active covering weights
      1. if $\theta \leq \delta\eta\langle e_i, By\rangle/\varepsilon$
         // approximate $w_i \leftarrow \exp(-\delta\eta\langle e_i, By\rangle)$
         a. $w_i \leftarrow \exp(-\varepsilon)w_i$
         b. if $w_i \leq \exp(-\eta)$
            // i made inactive if weight small enough
            1. $\mathcal{Q} \leftarrow \mathcal{Q} - i$
6. return $x$
```

Figure 3: *A randomized, width-independent implementation of the MWU framework from [41] specialized to a pure covering problem of the form* $\min\langle c, x\rangle$ *s.t* $Bx \geq \mathbb{1}, x \geq \mathbb{0}$.

$O\big(N + m\ln(n)/\varepsilon^2\big)$ *time. Each packing weight* $v_i, i \in \mathcal{P}$ *increases along integral powers of* $\exp(\varepsilon)$ *from 1 to (at most)* $\exp(\ln(m_p)/\varepsilon)$. *Each covering weight* $w_i, i \in \mathcal{C}$ *decreases along integral powers of* $\exp(\varepsilon)$ *from 1 to* $\exp(-\ln(m_c)/\varepsilon)$.

The $O(N)$ term comes from examining each nonzero of A and B in order to prepare for the randomized update step in lines (`5.G.i`) and (`5.H.i`). In geometric instances, the number of nonzeroes in A and B may be large, and N could be quadratic in the input size $m + n$. Next we show how to avoid the dependence on N in the geometric setting via data structures.

The second component appropriated from [41] is partially dynamic randomized data structure that maintains relative coordinatewise approximations of the vectors $A^T v$ and $B^T w$ as v increases and w decreases when A and B are given explicitly. This data structure will be used to handle the explicit "side constraints" given by A' and B' in the statements of Theorem 2.1 and Theorem 2.2. A simplified version of the result in [41] that suffices for our setting is as follows.

Theorem 3.2. ([41]) *Let* $\varepsilon > 0$ *and* $W > 1$ *be fixed and known.*

(i) Let $A \in \mathbb{R}_{\geq 0}^{m \times n}$ *be a nonnegative matrix with* N *nonzero coordinates all in the range* $[2^{-\widetilde{O}(1)}, 2^{\widetilde{O}(1)}]$ *and* $v \in \mathbb{R}_{>0}^m$ *be a positive vector initialized to* $\mathbb{1}$ *and incremented online by* T, *such that* $v_j \leq W$ *for all* j *at all times. Then one can maintain an* $(1 \pm \varepsilon)$*-multiplicative approximation of each coordinate of* $A^T v$ *in with high probability in* $\widetilde{O}\left(T + N\log W + \frac{n\log W}{\varepsilon^2}\right)$ *total randomized time.*

(ii) Let $B \in \mathbb{R}_{\geq 0}^{m \times n}$ *be a nonnegative matrix with* N *nonzero coordinates all in the range* $\left[2^{-\widetilde{O}(1)}, 2^{\widetilde{O}(1)}\right]$, *and let* $w \in \mathbb{R}_{\geq 0}^m$ *be a nonnegative vector initialized*

to $\mathbb{1}$ and decremented online by T single coordinate updates, such that $w_j \in [1, 1/W] \cup \{0\}$ at all times. Then one can maintain an $(1 \pm \varepsilon)$-multiplicative approximation of each coordinate of $B^T w$ in with high probability in $\widetilde{O}\left(T + N \log W + \frac{n \log W}{\varepsilon^2}\right)$ total randomized time.

4. Fast Implementation of `random-mwu` for Geometric Problems

We describe how (randomized) geometric data structures can be used to develop fast implementations of `random-mwu` in various settings. We focus on the most general setting of mixed packing and covering from Theorem 2.2. Recall that in this setting we wish to solve a positive LP of the form

$$\begin{aligned} &\text{find } x \in \mathbb{R}^n_{\geq 0} \\ &\text{s.t. } Ax \leq \mathbb{1},\ B'x \leq \mathbb{1},\ Bx \geq \mathbb{1}, \text{ and } C'x \geq \mathbb{1}, \end{aligned}$$

where $A, B \in \mathbb{R}^{m\times n}_{\geq 0}$ are associated with a geometric range space (P, R) with $n = |R|$ and $m = |P|$, and B' and C' are general non-negative matrices that model side constraints and are given explicitly while A, B are implicitly specified. Let $A = \operatorname{diag}(\alpha^A) I(P, R) \operatorname{diag}(\beta^A)$ and $B = \operatorname{diag}(\alpha^B) I(P, R) \operatorname{diag}(\beta^B)$ where $I(P, R)$ is incidence matrix of (P, R).

The two key components in the implementation are: (i) updating the weights of the constraints in each iteration (ii) solving the simplified optimization problem in each iteration. The next two subsections show how to efficiently handle the implicit matrices A, B. We put together the details for the overall running time in the final subsection.

4.1. Implementing randomized weight updates via range searching We address the efficient implementation of the weight update step in `random-mwu` hen we are working with an positive LP defined by a range space (P, R). This consists of finding the step size δ in (`5.C`) and updating the weights of the packing constraints and covering constraints (steps (`5.G.i`) and (`5.H.i`)). We first focus on the weight update process. We focus on updating implicit packing weights since updating the covering weights is similar. Recall that in the general setting, the algorithm picks a random θ and updates the weights of all rows i such that $\theta \leq \delta\eta\langle e_i, Ay\rangle/\varepsilon$. In the implicit setting, for a range space (P, R), A is of the form $A = \operatorname{diag}(\alpha) I \operatorname{diag}(\beta)$ where $\alpha \in \mathbb{R}^P_{\geq 0}$ and $\beta \in \mathbb{R}^R_{\geq 0}$, and y is of the form γe_r for some $r \in R$. Substituting in, we have that for any $p \in P$, we update the weight for p iff $p \in r$ *and* $\theta \leq \delta\eta\alpha_p\beta_r/\varepsilon$. That is, iff $p \in r$ and $\alpha_p \geq \frac{\theta\varepsilon}{\delta\eta\beta_r}$. Crucially, the threshold $\frac{\theta\varepsilon}{\delta\eta\beta_r}$ against which we compare α_p is independent of the weight of p. Thus the weight update reduces to the following search problem: given a range $r \in R$ and a threshold τ, find all the points in $p \in P \cap r$ such that $\alpha_p \geq \tau$. We emphasize that α_p is static and does not change over the course of the algorithm. This is essentially range search, formalized below.

Definition 4.1. *Let (P, R) be a range space. A* range search *takes as input a range $r \in R$ and returns $P \cap r$. Given* static *values $\mu : P \to \mathbb{R}$, an* interval range search *takes as input a range $r \in R$ and an interval $[a, b] \subseteq \mathbb{R}$, and returns the set $\{p \in P \cap r : a \leq \mu(p) \leq b\}$ A* range search with deletions *(resp. interval range search with deletions) data structure allows points to be deleted from the underlying point set P.*

Very efficient range search data structures are known for various geometric settings. For instance when the ranges are d-dimensional axis-aligned boxes in $\mathbb{R}^d$, range search with query time $\log^{O(d)} n$ can be achieved via relatively simple data structures. Here we are assuming only emptiness oracles and very little else. It is well-known that range search with static values can be implemented via emptiness oracles with only logarithmic overhead. (One can improve the logarithmic factors in more specified setting, but we abstain from these details per the discussion before Notation 2.1.) We describe it below for sake of completeness.

Lemma 4.1. *Given an emptiness oracle (resp. emptiness oracle with deletions) data structure for (P, R), one can implement an interval range search (resp. interval range search with deletions) data structure for static values μ where the overhead per query is $O(\log m)$.*

Proof. We sort the points P in increasing order of their μ values and build a balanced binary tree T over P. For each internal node v in the tree T, build an emptiness oracle data structure over all the points in the subtree T_v rooted at v. Each interval range query $(r, [a, b])$ decomposes into queries of (unweighted) range reporting queries with range r at a logarithmic number of nodes in T. For each such node v we can use the emptiness oracle data structure at v (and within T_v) to report all the points in T_v contained in r in time proportional to number of points reported (this is standard: if r has no points in T_v we stop otherwise we recursively explore the two children of v). Thus the total time is $O((k+1) \log m)$ queries to the emptiness oracle where k is the total number of points reported. To delete a point p we remove it from the emptiness oracle data structure at $\Theta(\log m)$ levels in which p participates so that future queries are handled correctly. We note that the tree T is unchanged even though points are removed from the emptiness oracle data structures. ∎

The preceding lemma with the observation at the start of the subsection allows us to efficiently (assuming efficient emptiness oracle data structure) list all the points p in a range r whose packing and covering weights need to be updated after θ is chosen randomly.

We address the issue of choosing the step size δ. Suppose we have chosen to update x by the single coordinate solution $y = \delta e_r$ for some range r. Note that δ is chosen in `(5.C)`. To determine δ in the implicit setting we need to find the bottleneck point $p \in r$ with the largest value of α_p. We can find this from the same data structure as in Lemma 4.1 in $\widetilde{O}(1)$ time as follows. Recall that T stores P in sorted order of α_p values. We start at the root of the tree T and explore the left and right child of the root to see if r has any points in the subtrees. Since T stores points in increasing order of α values, we recursively explore the right child if r is non-empty on that side, otherwise we recursively explore the left child. It is easy to see that in $O(\log m)$ queries we will find the desired point.

The preceding discussion leads to the following lemma that captures the total time spent on updating weights.

Lemma 4.2. *Let $\varepsilon > 0$ be given. Consider a positive LP with m_p explicit packing constraints and m_c explicit covering constraints, in addition to packing and covering constraints induced by a range space (P, R). Let the explicit packing and covering constraints be given by N nonzeroes.*

(a) If there are only induced packing constraints, and (P, R) has nearly linear emptiness oracles, then modulo the time spent in lines `(5.A)` and `(5.C)`, `random-mwu` can be implemented in $\widetilde{O}\left(N + \frac{m}{\varepsilon^2}\right)$ total time.

(b) If (P, R) has nearly linear emptiness oracles with deletions, then modulo the time spent in lines `(5.A)` and `(5.C)`, `random-mwu` can be implemented in $\widetilde{O}\left(N + \frac{m}{\varepsilon^2}\right)$ total time.

In either case, the packing weights are increased monotonically along integer powers of $\exp(\varepsilon)$ from 1 to $m^{O(1/\varepsilon)}$, and the covering weights are decreased monotonically along integer powers of $\exp(-\varepsilon)$ from 1 to $m^{-O(1/\varepsilon)}$.

Proof. We apply Theorem 3.1 to an implementation where the randomized weight updates are implemented by range search data structure. By Lemma 4.1, with $\widetilde{O}(m)$ initialization time, we can implement the sampling steps in lines `(5.G.i)` and `(5.H.i)` in $\widetilde{O}(1)$ time per point listed in the sample. Each point listed has its corresponding weight updated at most $\widetilde{O}\left(\frac{1}{\varepsilon^2}\right)$ times, so the total time spent on sampling weights of induced constraints is $\widetilde{O}(m/\varepsilon^2)$. We note that when an induced covering constraint corresponding to a point $p \in P$ is dropped in line `(5.H.i.b)`, the point p has to be removed from the range search data structure. If the emptiness oracles have $\widetilde{O}(1)$ time deletions, then p can be removed from all future weight updates in $\widetilde{O}(1)$ time as described in Lemma 4.1. The desired running times then follow from Theorem 3.1. ∎

4.2. Implementing the greedy oracle (approximately) with emptiness oracles

Lemma 4.2 reduces positive linear programs defined implicitly over range spaces with nearly linear emptiness oracles to a fast implementation of line `(5.A)`. In particular, to obtain nearly linear running times, we need to implement `(5.A)` in $\widetilde{O}(1)$ amortized time. The subproblem solved in `(5.A)` is relatively simple compared to the original positive linear program and the basic set up is as follows. We let vectors v and v' denote the weight vectors maintained by the MWU algorithm for the packing constraints defined by A, A' respectively, and w, w' denote the weight vectors maintained by the algorithm for the covering constraints defined by B, B' respectively. The packing weights increase monotonically and the covering weights decrease monotonically as the corresponding constraints tighten. In each iteration, we need to compute a range $r \in R$ that approximately maximizes the ratio $\frac{\langle w', B' e_r\rangle + \langle w, B e_r\rangle}{\langle v', A' e_r\rangle + \langle v, A e_r\rangle} = \frac{\langle w', B' e_r\rangle + \beta_r^B \sum_{p \in r} \alpha_p^B w_p}{\langle v', A' e_r\rangle + \beta_r^A \sum_{p \in r} \alpha_p^A v_p}$. We note that whenever an induced covering constraint for a point $p \in P$ is deleted in line `(5.H.i.b)`, we decrease w_p to 0.

The main technical challenge is to handle *decreasing* weights. We overcome this challenge in our setting via a careful analysis of the requirements of the randomized MWU algorithm.

4.2.1. Depth estimation

We first consider a simpler setting with just one set of weights $w : P \to \mathbb{R}_{\geq 0}$. A basic and seemingly necessary task is approximating the weight $\sum_{p \in r} w_p$ of any range $r \in R$. Formally, the setup is as follows.

Setting 4.1. *Let (P, R) be a range space with m points and n ranges and equipped with nearly linear emptiness* *oracles. Let P be weighted by $w : P \to \mathbb{R}_{\geq 0}$. For $r \in R$, let $\overline{w}(r) = \sum_{p \in r} w_p$ denote the sum weight with respect to w of points in r. We assume that the weight $\overline{w}(r)$ of any range $r \in R$ is always either 0 or in the interval $[1/W, W]$ for a fixed and known $W > 0$.*

For some geometries, such as intervals on a line or rectangles in the plane, one can compute and maintain the total weight of each range in polylogarithmic time. Other basic settings such as disks (and in particular, those without well-behaved "canonical sets") are known to require polynomial time to maintain weights exactly. When it comes to relative approximations, however, [15] observed that nearly linear emptiness oracles suffice to efficiently estimate the weight by sampling.

Fact 4.1. ([15]) *There is a randomized algorithm that builds a data structure in $\widetilde{O}\left(\frac{m \log \log W}{\varepsilon^3}\right)$ time which can estimate the weight of any range $r \in R$ to within an $(1 \pm \varepsilon)$-multiplicative factor in $\widetilde{O}\left(\frac{\log \log W}{\varepsilon^2}\right)$ time. The algorithm succeeds with high probability.*

Note that the above is for a given set of static weights. Applying the standard dynamization techniques of Bentley and Saxe [1], one obtains the following partially dynamic guarantees when the weights are *increasing*.

Fact 4.2. *Suppose w is updated online by $T = \operatorname{poly}(m, n)$ single-coordinate weight increments. There is a randomized data structure that in $\widetilde{O}\left(\frac{(m+T) \log \log W}{\varepsilon^3}\right)$ total time estimates the weight $\overline{w}(r)$ of any given range $r \in R$ to within an $(1 \pm \varepsilon)$-multiplicative factor in $\widetilde{O}\left(\frac{\log \log W}{\varepsilon^2}\right)$ time. All the estimates are correct with high probability even for adaptive queries.*

4.2.2. Maintaining the maximum ratio range

We return now to implementing `random-mwu`. To implement (5.A), it suffices to find a range $r \in R$ that approximately maximizes the ratio of two sets of weights, $\frac{\langle w', B'e_r \rangle + \beta_r^B \sum_{p \in r} \alpha_p^B w_p}{\langle v', A'e_r \rangle + \beta_r^A \sum_{p \in r} \alpha_p^A v_p}$

To model this, we consider the following setting

Setting 4.2. *Let (P, R) be a range space with m points and n ranges and equipped with nearly linear emptiness oracles. Let P be weighted by two different sets of weights $v : P \to \mathbb{R}_{>0}$ and $w : P \to \mathbb{R}_{\geq 0}$. Let R be weighted by $w'' : R \to \mathbb{R}_{\geq 0}$ and $v'' : R \to \mathbb{R}_{>0}$. Let $\beta^A \in \mathbb{R}^R_{>0}$ and $\beta^B \in \mathbb{R}^R_{>0}$. For a fixed and known $W > 0$, we assume that the weight of any point with respect to v or range with respect to v' is always in the interval $[1/W, W]$, and the weight of any point with respect to w or range with respect to w' is always either 0 or in the interval $[1/W, W]$.*

In the preceding setting, v'' and w'' in Setting 4.2 will correspond to approximations of $(A')^T v'$ and $(B')^T w'$ maintained by Theorem 3.2. We abuse notation slightly and use $\overline{v}(r)$ to denote the quantity $\sum_{p \in r} \alpha_p^A v_p$ and $\overline{w}(r)$ to denote the quantity $\sum_{p \in r} \alpha_p^B w_p$

Lemma 4.3. *Suppose w and w' are decreased online and v and v' are increased online by $T = \operatorname{poly}(m, n)$ single-coordinate weight updates. Let $\beta^A, \beta^B \in \mathbb{R}^R_{>0}$ be two fixed vectors with coordinates in the range $\left[2^{-\widetilde{O}(1)}, 2^{\widetilde{O}(1)}\right]$. One can maintain, with high probability in $\widetilde{O}\left(\frac{((m+n) \log W + T) \log \log W}{\varepsilon^3}\right)$ total time, a range $r \in R$ with weighted ratio $\frac{w''_r + \beta_r^B \overline{w}(r)}{v''_r + \beta_r^A \overline{v}(r)}$ within a $(1 \pm \varepsilon)$-multiplicative factor of the maximum ratio range.*

Proof. Let $\widetilde{W} = 2^{\widetilde{O}(1)} W$. By assumption, the maximum ratio $\max_{r \in R} \frac{w''_r + \beta_r^B \overline{w}(r)}{v''_r + \beta_r^A \overline{v}(r)}$ is either 0 or in the range $[\widetilde{W}^2, 1/\widetilde{W}^2]$ at any point in time, and is monotonically decreasing as w and w' decrease and v and v' increase. By Fact 4.2, we can maintain a $(1 \pm \varepsilon)$-multiplicative estimate of $\overline{v}(r)$ for all $r \in R$ in the allotted time and space. Maintaining $\overline{w}(r)$ is not straightforward because w is *decreasing*. However, for the sake of maintaining the maximum ratio range (up to an $(1 \pm O(\varepsilon))$-multiplicative factor) we can apply thresholding techniques to reduce decremental weight estimation to a more feasible decision problem as follows.

We maintain a threshold λ such that $\lambda \geq \max_{r \in R} \frac{w''_r + \beta_r^B \overline{w}(r)}{v''_r + \beta_r^A \overline{v}(r)}$. λ is initialized to $\widetilde{W}^2$ and decreased by powers of $(1 - \varepsilon)$. At any point, we either output a range $r \in R$ with ratio at least $(1 - O(\varepsilon))\lambda$, or certify that there is no range with ratio at least $(1 - \varepsilon)\lambda$. In the latter case, we replace λ with $(1 - \varepsilon)\lambda$. Note that λ decreases at most $\widetilde{O}\left(\frac{\log W}{\varepsilon}\right)$ times before it falls out of the range $[1/\widetilde{W}^2, \widetilde{W}^2]$.

Fix a value λ. We claim that it suffices to implement the following query efficiently: given $r \in R$, decide if its ratio is $\leq (1 - \varepsilon)\lambda$, or its ratio is $\geq (1 - O(\varepsilon))\lambda$. Indeed, given such a query, we test each $r \in R$ in (any) fixed order until we find a range r with ratio at least $(1 - O(\varepsilon))\lambda$. We pause the search at this range and output it. As the algorithm proceeds and the weights of the constraints change, the ratios of ranges are affected. Note, however, that the ratio of any range monotonically decreases. Thus, we can keep updating and using the currently held range r as long as its ratio is at least $(1 - O(\varepsilon))\lambda$. Once its ratio falls below $(1 - \varepsilon)\lambda$ we resume the search for a new range among the ranges in the fixed order (continuing from r), until we either find another range with ratio at least $(1 - O(\varepsilon))\lambda$, or declare that all ranges have ratio $< (1 - \varepsilon)\lambda$. Since the ratio of any range is monotonically decreasing, after rejecting a range once for a fixed λ, we do not need to inspect it again.

Thus, for a fixed upper bound λ on the ratio of any range, we want to decide, for each range r, whether the ratio of r is $\leq (1 - \varepsilon)\lambda$ or $\geq (1 - O(\varepsilon))\lambda$. (Note that the overlap allows for either decision if the ratio lies between $(1 - \varepsilon)\lambda$ and $(1 - O(\varepsilon))\lambda$.) The technical difficulty lies in the fact that the weights $w : P \to \mathbb{R}_{>0}$ are decreasing, while Fact 4.2 can only maintain estimates for increasing weights. To circumvent this, we periodically take a snapshot w^0 of w, and track the *increasing* difference $w^0 - w$. Intuitively, if w^0 is reset frequently enough, then $(1 \pm \varepsilon)$-multiplicative estimates for w^0 and $w^0 - w$ sum to sufficiently good multiplicative estimates for $w = w^0 + (w^0 - w)$.

More precisely, we set (and maintain) an index $\ell = \lceil \log \lambda \rceil$. ℓ is initially $\lceil \log W^2 \rceil$ and decreases at most $O(\log W)$ times. Whenever ℓ is set or reset, we build the following data structures. w^0 denotes the values of the weights w when ℓ is decreased. By Fact 4.1, we can compute $(1 \pm \varepsilon)$-multiplicative approximations with respect to w^0 in $\widetilde{O}\left(\frac{m \log \log W}{\varepsilon^3}\right)$ time. By Fact 4.2, until ℓ decreases, we can maintain an $(1 \pm \varepsilon)$-multiplicative approximation with respect to $w^0 - w$ in $\widetilde{O}\left(\frac{T_\ell \log \log W}{\varepsilon^3}\right)$ total time, where T_ℓ is the total number of increments until ℓ decreases.

Now, given $r \in R$, we decide if the ratio of r is at least $(1 - O(\varepsilon))\lambda$ or $(1 - \varepsilon)\lambda$ as follows. Using $(1 \pm \varepsilon)$-multiplicative estimates for w^0, $w^0 - w$, and v, let $\rho \in (1 \pm \varepsilon)\big(w''_r + \beta^B_r \overline{w}^0(r)\big)$, $\sigma \in (1 \pm \varepsilon)\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)$, and $\tau \in (1 \pm \varepsilon)\big(v''_r + \beta^A_r \overline{v}(r)\big)$ (all with high probability). If $\frac{\rho - \sigma}{\tau} \geq (1 - O(\varepsilon))\lambda$ (for a sufficiently large constant hidden in $O(\varepsilon)$), we decide that r has ratio at least $(1 - O(\varepsilon))\lambda$ (for a slightly larger hidden constant); otherwise if $\frac{\rho - \sigma}{\tau} \leq (1 - O(\varepsilon))\lambda$, we decide that r has ratio less than $(1 - \varepsilon)\lambda$.

We claim that, if $\rho \in (1 \pm \varepsilon)\big(w''_r + \beta^B_r \overline{w}^0(r)\big)$, $\sigma \in (1 \pm \varepsilon)\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)$, and $\tau \in (1 \pm \varepsilon)\big(v''_r + \beta^A_r \overline{v}(r)\big)$, then the decision is correct. To this end, we first observe that since

$$\frac{\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)}{v''_r + \beta^A_r \overline{v}(r)} \leq \frac{w''_r + \beta^B_r \overline{w}^0(r)}{v''_r + \beta^A_r \overline{v}(r)} \leq O(\lambda),$$

we have

$$\begin{aligned}\frac{\sigma}{\tau} &\leq (1 + O(\varepsilon))\frac{\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)}{v''_r + \beta^A_r \overline{v}(r)} \\ &= \frac{\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)}{v''_r + \beta^A_r \overline{v}(r)} + O(\varepsilon\lambda),\end{aligned}$$

Thus, if $\frac{\rho - \sigma}{\tau} \geq (1 - O(\varepsilon))\lambda$, we have

$$\begin{aligned}\frac{w''_r + \beta^B_r \overline{w}(r)}{v''_r + \beta^A_r \overline{v}(r)} &= \frac{w''_r + \beta^B_r \overline{w}^0(r)}{v''_r + \beta^A_r \overline{v}(r)} - \frac{\beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)}{v''_r + \beta^A_r \overline{v}(r)} \\ &\geq (1 - O(\varepsilon))\frac{\rho}{\tau} - \frac{\sigma}{\tau} - O(\varepsilon\lambda) \\ &\geq (1 - O(\varepsilon))\frac{\rho - \sigma}{\tau} - O(\varepsilon\lambda) \\ &\geq (1 - O(\varepsilon))\lambda,\end{aligned}$$

as desired. Conversely, if $\frac{w''_r + \beta^B_r \overline{w}(r)}{v''_r + \beta^A_r \overline{v}(r)} \geq (1 - \varepsilon)\lambda$, then

$$\begin{aligned}\frac{\rho - \sigma}{\tau} &\geq (1 - O(\varepsilon))\frac{w''_r + \beta^B_r \overline{w}^0(r)}{v''_r + \beta^A_r \overline{v}(r)} \\ &\quad - \frac{w''_r + \beta^B_r\big(\overline{w}^0(r) - \overline{w}(r)\big)}{v''_r + \beta^A_r(\overline{v}(r))} - O(\varepsilon\lambda) \\ &\geq (1 - O(\varepsilon))\frac{w''_r + \beta^B_r \overline{w}(r)}{v''_r + \alpha_r \overline{v}(r)} - O(\varepsilon\lambda) \\ &\geq (1 - O(\varepsilon))\lambda,\end{aligned}$$

as desired. Thus we can implement the overlapping decision problem correctly with high probability, and thus maintain an approximately maximum ratio range, as desired. We now analyze the total time. Let T_λ be the total number of weight estimate queries with respect to w for a fixed value of λ; we have $\sum_\lambda T_\lambda = T$.

- $\widetilde{O}\left(\frac{(m + T)\log \log W}{\varepsilon^3}\right)$ time for maintaining an $(1 \pm \varepsilon)$-multiplicative factor with respect to v;
- $\widetilde{O}\left(\frac{(m + T_\ell)\log \log W}{\varepsilon^3}\right)$ time for building and maintaining estimates for w^0 and $w^0 - w$ for each fixed ℓ, and
- $\widetilde{O}\left(\frac{(n + T_\lambda)\log \log W}{\varepsilon^2}\right)$ time for querying weight estimates for ranges with respect to w for fixed λ.

Index ℓ takes $O(\log W)$ different values and λ takes $O\left(\frac{\log W}{\varepsilon}\right)$ different values, and $\sum_\lambda T_\lambda \sum_\ell T_\ell \leq T$, so summing the above together, the total time is $\widetilde{O}\left(\frac{((m+n)\log W + T)\log\log W}{\varepsilon^3}\right)$, as desired. ∎

Remark 4.1. *A careful reader may wonder where in the preceding proof did we need emptiness oracle for deletions. In the setting of Theorem 2.1 we only have packing constraints associated with (P, R) and packing weights only increase and no points are ever deleted. However, when there are covering constraints $Bx \leq \mathbb{1}$ associated with (P, R), we need to maintain decreasing weights and a point corresponding to a covering constraint is made inactive when its weight becomes too low. We implemented decreasing weights by using snapshots and rebuilding data structures that handle increments. Once a point p is dropped for covering, it does not participate in those rebuilt data structures. Thus we use the emptiness oracle for deletions only in Section 4.1 where we need to report active points in a range for updating the weights.*

4.3. Putting together the implementation of `random-mwu`

We now complete the proofs of Theorem 2.1 and Theorem 2.2.

Proof. (Proofs of Theorem 2.1 and Theorem 2.2) By Lemma 4.2, it suffices to implement lines (5.A) and (5.C) in $\widetilde{O}(1)$ amortized time. We implement lines (5.A) to (5.C) via Lemma 4.3 as follows. By Theorem 3.2, we can maintain $(1 \pm \varepsilon)$-multiplicative approximations to each coordinate of $(A')^T v$ and $(B')^T w$ in the allotted time. By Lemma 4.3, taking α_R as α, β_R as β, $\alpha_p^B w_{P,p}$ as w_p for each p, the $(1 \pm \varepsilon)$-multiplicative approximation of $(A')^T w'$ as w', $\alpha_p^A v_p$ as v_p for each p, and the $(1 \pm \varepsilon)$-multiplicative approximation $(A')^T v'$ as v', we can maintain a range r with weighted ratio

$$\frac{\langle w', B'e_r\rangle + \langle w, Be_r\rangle}{\langle v', A'e_r\rangle + \langle v, Ae_r\rangle} = \frac{\langle w', B'e_r\rangle + \beta_{R,r}\sum_{p\in R}\alpha_p^B w_p}{\langle v', A'e_r\rangle + \alpha_{R,r}\sum_{p\in R}\alpha_{R,r} v_p}$$

within a $(1 \pm O(\varepsilon))$-multiplicative factor of the maximum ratio range. With $(1 \pm \varepsilon)$-multiplicative approximations for $\sum_{p\in R}\alpha_p^A v_p$, we can also choose $\gamma > 0$ such that $\gamma\alpha_{R,r}\sum_{p\in r}\alpha_p^A v_p \in (1 \pm \varepsilon)\sum_{p\in P} v_p$. It is easy to see then that $y = \gamma e_r$ satisfies the conditions of line (5.A). For y of this form, (5.C) can be implemented (approximately) by range search data structures to find the "bottleneck constraint" and set δ accordingly, by the same construction as when implementing the random sampling step in Section 4.1.

By Theorem 3.1, we have $T = \widetilde{O}(m/\varepsilon^2)$ and $W = m^{\widetilde{O}(1/\varepsilon)}$. Plugging into the bounds of Lemma 4.3 gives us the time complexity we seek. ∎

5. Implicit range spaces

Section 4.2 shows how to solve positive linear programs defined implicitly over range spaces in nearly linear time, even when there are $\Omega(mn)$ incidences between points and ranges. In this section, we consider an even more difficult setting where not only is the incidence structure a large polynomial in the input size, but even the range space is a large polynomial in the input size. A motivating example of this situation is the following.

Example 5.1. *Given n disks D in the plane weighted by $c : D \to \mathbb{R}_{>0}$, consider the following LP for computing the maximum (weight) independent set.*

$$\text{maximize } \sum_D c_D x_D \text{ over } x : D \to \mathbb{R}_{\geq 0}$$
$$\text{s.t. } \sum_{D\ni p} x_D \leq 1 \text{ for all } p \in \mathbb{R}^2.$$

The above gives packing constraints continuously throughout the plane, but they can be discretized by listing the $m = O(n^2)$ distinct points in their arrangement. The first problem here, from the standpoint of obtaining nearly linear running times, is that the number of points in the implicit range space is still quadratic, let alone the fact that the incidence matrix may be dense.

In this section, we consider problems in the following formal setting.

Definition 5.1. *Let R be a set of objects in the plane. The* union complexity *$u : \mathbb{N} \to \mathbb{N}$ of R is the function mapping $n \in \mathbb{N}$ to the maximum number of arcs on the boundary of the union of any n objects in R.*

Setting 5.1. *Let $(\mathbb{R}^2, R)$ be a range space consisting of $n = |R|$ ranges such that*

(a) The dual range space $(R, \mathbb{R}^2)$ has nearly linear emptiness oracles with deletions.

(b) R has nearly linear union complexity $u(n) = \widetilde{O}(n)$.

Let R be equipped with positive weights $c : R \to \mathbb{R}_{>0}$.

We note that disks and fat triangles in the plane satisfy the above conditions. Pseudo disks have linear sized union complexity. We also need the range space $(R,^2)$ to have nearly linear emptiness oracle with deletions. For pseudo disks this is also possible to achieve via known techniques [38].

Given Setting 5.1, we consider the problem of computing the maximum weight fractional independent set:

$$\begin{aligned} \text{maximize } & \sum_{r \in R} c_r x_r \text{ over } x : R \to \mathbb{R}_{\geq 0} \\ \text{s.t. } & \sum_{r \ni p} x_r \leq 1 \text{ for all } p \in \mathbb{R}^2. \end{aligned}$$

To this end, we apply the MWU framework to the associated dual problem:

$$\begin{aligned} \text{minimize } & \sum_{p \in \mathbb{R}^2} x_p \text{ over } x : \mathbb{R}^2 \to \mathbb{R}_{\geq 0} \\ \text{s.t. } & \sum_{p \in r} x_p \geq c_r \text{ for all } r \in R. \end{aligned}$$

The above is a hitting set problem, where we want to compute a minimum cardinality (fractional) set of points hitting each disk. Not only does the MWU framework give an $(1 \pm \varepsilon)$-multiplicative approximation to the above (dual) problem, but standard techniques (see, e.g., [35]) can extract a $(1 \pm \varepsilon)$-multiplicative approximation to the original (primal) problem from the evolving sequence of weights w, as desired.

By Theorem 3.1 and Lemma 4.2, `random-mwu` with randomized weight updates implemented by emptiness oracles gives an algorithm that takes $\widetilde{O}(n/\varepsilon^2)$ time plus $\widetilde{O}(n/\varepsilon^2)$ calls to an oracle in line (5.A) that, for this particular case of pure covering, reduces to the following subproblem:

$$\begin{aligned} \text{maximize } & \sum_{p \in \mathbb{R}^2} x_p \text{ over } x : \mathbb{R}^2 \to \mathbb{R}_{\geq 0} \\ \text{s.t. } & \sum_{r \in R} \frac{w_r}{c_r} \sum_{p \in r} x_p \geq \sum_{r \in R} w_r, \end{aligned}$$

where $w : R \to \mathbb{R}_{\geq 0}$ are nonnegative weights on R each decreasing monotonically from 1 to $m^{-O(\varepsilon)}$ and then to 0. This subproblem can be solved (approximately) by finding the point p with (approximately) maximum depth $\sum_{r \ni p} \frac{w_r}{c_r}$, and taking $y = \gamma e_r$ for $\gamma = \frac{\sum_{r \in R} w_r}{\sum_{r \ni p} \frac{w_r}{c_r}}$. The goal, then, is to approximate the deepest point with respect to the weights w_r/c_r in $\widetilde{O}(\text{poly}(1/\varepsilon))$ amortized time per iteration.

5.1. Approximating the deepest point in $\widetilde{O}(1/\varepsilon^2)$ amortized time We want to approximate the deepest point in a weighted set of regions, in time faster then listing all the possible intersection points. To this end, we apply sampling techniques that leverage the underlying geometry to generate a comprehensive list of $\widetilde{O}(n)$ points to test. The machinery for this sampling-based approach is built on well-known techniques in computational geometry; see [19] for a more general overview and other applications of these techniques.

Definition 5.2. *Let R be a set of ranges in the plane with total weight W, and $\varepsilon \in \{0, 1\}$. An ε-cutting is a decomposition C of the plane into regions such that (a) the number of regions in C is small and (b) for $c \in C$, the total weight of ranges $r \in R$ whose boundary intersects the interior of c is at most εW.*

Fact 5.1. ([2]) *Consider a family of ranges with union complexity u. Then the number of vertices induced by n ranges with (unweighted) depth $\leq k$ is at most $O(k^2 u(n/k))$.*

Lemma 5.1. *Let R be a collection of n ranges in the plane with nearly linear union complexity $u(k) = \widetilde{O}(k)$ and weighted by $w : R \to \mathbb{R}_{>0}$. Let $W = \sum_{r \in R} w_r$ be the total weight, and let $D \geq \max_{p \in \mathbb{R}^2} \sum_{r \ni p} w_r$ be an upper bound on the maximum weighted depth of D. The one can compute, with high probability in $\widetilde{O}(n)$ randomized time, a set of $\widetilde{O}(n)$ points $P \subseteq \mathbb{R}^2$ such that, for any subset of ranges $S \subseteq R$ with nonempty intersection and total weight $\overline{w}(S) \geq D/4$, we have $P \cap \left(\bigcap_{r \in S} r\right) \neq \emptyset$.*

The constant $1/4$ is arbitrary, and chosen for convenience.

Proof. Let $\varepsilon = \frac{D}{4W}$, and note that $\frac{1}{\varepsilon} = O\left(\frac{W}{D}\right) \leq O\left(\frac{W}{\max_{r \in R} w_r}\right) = O(n)$. Let $S \subseteq R$ sample each range $r \in R$ independently with probability $\widetilde{O}\left(\frac{w_r}{\varepsilon W}\right) = \widetilde{O}\left(\frac{w_r}{D}\right)$. S has cardinality $|S| = \widetilde{O}\left(\frac{1}{\varepsilon}\right) = \widetilde{O}(n)$ with high probability. [42, Lemma 3.9] implies that the vertical decomposition A induced by S is an ε-cutting with respect to R. Moreover, by combining the union bound with the Chernoff inequality, S has (unweighted) depth at most $\widetilde{O}(1)$ with high probability. As such, a sweeping construction of A takes $\widetilde{O}(|S|) = \widetilde{O}(n)$ time. By Fact 5.1, since R (hence S) has nearly linear union complexity, and S has depth $\widetilde{O}(1)$, the vertical decomposition A of S has $\widetilde{O}(|S|) = \widetilde{O}(n)$ vertices V. We claim that, since A is ε-cutting, any subset $T \subseteq R$ with nonempty intersection and total weight $\sum_{r \in T} w_r \geq D/4 = \varepsilon W$ had $V \cap \left(\bigcap r \in T\right) \neq \emptyset$. To this end, we first observe that for any region a in the vertical decomposition A, vertex v of a, point $p \in a$ that is not a vertex of a, and range $r \in R$, we have $p \in r$ and $a \notin r$ iff the boundary of r properly crosses the interior of a. (This follows from our assumption on the general position of R.) Now, let $T \subseteq R$ have total weight at least εW and $p \in \bigcap_{r \in T} r$. Then p is in some region a of A. If $\bigcap_{r \in T} r$ does not contain any vertices of A, then in particular p is not a vertex of a, and every range $r \in T$ misses some vertex of a. By the above claim, then, every $r \in T$ has boundary intersecting the interior of a. But then the sum of ranges whose boundary intersects the interior of a is at least εW, a contradiction to the fact that A is an ε-cutting. By contradiction, then, we must have $V \cap \left(\bigcap_{r \in T} r\right) \neq \emptyset$, as desired. □

Setting 5.2. *Continuing Setting 5.1, let R have nonnegative weights $w : R \to \mathbb{R}_{\geq 0}$ decreased online by T single-weight updates. Let $W > 0$ be a known and fixed value such that for any range $r \in R$, we have $w_r \in \{0\} \cup [1/W, W]$. For $p \in \mathbb{R}^2$, let $\overline{w}(p) = \sum_{r \in R} w_r$ denote the weighted depth of p.*

Lemma 5.2. *One can maintain, in $\widetilde{O}\left(\frac{n \log W + T}{\varepsilon^3}\right)$ total time, a $(1 \pm \varepsilon)$-multiplicative approximation to the deepest point in the arrangement of R.*

Proof. There are two basic technical difficulties to overcome. The first, encountered earlier in Lemma 4.3, is that the weights w are monotonically decreasing, whereas Fact 4.2 only lets us maintain approximate depths when the weights are increasing. The second challenge, unique to this setting, is that the total number of candidate points in the explicit LP is on the order of the number of vertices in the arrangement of R, and a large polynomial in n. Even enumerating these points (let alone computing the arrangement) is too slow. We need to maintain the heaviest point without explicitly checking most of them.

Observe that as w decreases, the depth of any point is monotonically decreasing. Thus, we can apply thresholding strategies similar to Lemma 4.3 and reduce the problem instead to single-resolution decision problems, as follows.

We maintain a threshold $\lambda > 0$ with the invariant that $\lambda \geq \overline{w}(p)$ for all $p \in \mathbb{R}^2$. λ is initialized to $\widetilde{O}(W)$ and decreased by powers of $(1 - \varepsilon)$. At any point, we either maintain a point $p \in \mathbb{R}^2$ with $\overline{w}(p) \geq (1 - O(\varepsilon))\lambda$, or certify (with high probability) that $\overline{w}(p) < (1 - \varepsilon)\lambda$ for all points $p \in \mathbb{R}^2$. In the latter case, we replace λ with $(1 - \varepsilon)\lambda$, and continue. Note that λ decreases at most $O\left(\frac{\log W}{\varepsilon}\right)$ times before it falls out of the range $[\widetilde{O}(W), 1/\widetilde{O}(W)]$.

Let $\ell = \lceil \log \lambda \rceil$. ℓ is initialized to $\widetilde{O}(\log W)$, and decreases at most $\widetilde{O}(\log W)$ times. Whenever ℓ changes value (including initially), we do the following. Let w^0 denote the values of w when ℓ is set or reset. By Lemma 5.1, we generate a set of $\widetilde{O}(n)$ points $P_\ell \subseteq \mathbb{R}^2$ such that for any subset $S \subseteq R$ with nonempty intersection and total weight at least $\sum_{r \in S} w_r > 2^{\ell - 2}$, we have $P_\ell \cap \left(\bigcap_{r \in S} r\right) \neq \emptyset$. By Fact 4.1, we build a data structure giving $(1 \pm \varepsilon)$-multiplicative factor estimates of the depth of any point $p \in P_\ell$ with respect to w^0 in $\widetilde{O}\left(\frac{n \log \log W}{\varepsilon^3}\right)$. By Fact 4.2, we also initialize and maintain $(1 \pm \varepsilon)$-multiplicative factors of the depth of any point $p \in P_\ell$ with respect to the difference $w^0 - w$ in total time $\widetilde{O}\left(\frac{T_\ell \log \log W}{\varepsilon^3}\right)$.

We claim that that if $\overline{w}(p) < (1 - \varepsilon)\lambda$ for all $p \in P_\ell$, then $\overline{w}(q) < (1 - \varepsilon)\lambda$ for all $q \in \mathbb{R}^2$. Indeed, let $q \in \mathbb{R}^2$ with $\overline{w}(q) \geq (1 - \varepsilon)\lambda$. Let $S = \{r \in R : q \in r\}$ be the set of ranges containing q. S has nonempty intersection (containing q) and total weight $\sum_{r \in S} w_r = \overline{w}(q) \geq (1 - \varepsilon)\lambda \geq (1 - \varepsilon)2^{\ell - 1} > 2^{\ell - 2}$. By choice of P_ℓ, then, there exists $p \in P_\ell \cap \left(\bigcap_{s \in S} s\right)$. Such a point p has depth $\overline{w}(p) \geq \sum_{s \in S} \overline{w}(s) \geq (1 - \varepsilon)\lambda$, as desired.

Thus, for fixed λ, it suffices to maintain a point $p \in P_\ell$ with $\overline{w}(p) \geq (1 - O(\varepsilon))\lambda$, or certify that all $p \in P_\ell$ have depth $\overline{w}(p) < (1 - \varepsilon)\lambda$ to conclude that all points $q \in \mathbb{R}^2$ have depth $\overline{w}(q) < (1 - \varepsilon)\lambda$. This situation, by the same sweeping technique as in Lemma 4.3, reduces to the following overlapping decision problem: given $p \in P_\ell$, decide if $\overline{w}(p) < (1 - \varepsilon)\lambda$ or $\overline{w}(p) \geq (1 - O(\varepsilon))\lambda$. The same argument as in Lemma 4.3 shows that the $(1 \pm \varepsilon)$-multiplicative estimates for $\overline{w}^0(p)$ and $\overline{w}^0(p) - \overline{w}(p)$ suffice to implement the decision problem with high probability, and thus an $(1 \pm O(\varepsilon))$-multiplicative approximation of the deepest point in the arrangement. The total running time consists of

- $\widetilde{O}(n)$ to construct P_ℓ for each ℓ.
- $\widetilde{O}\left(\frac{(n + T_\ell) \log \log W}{\varepsilon^3}\right)$ time building and main-

taining estimates for w^0 and $w^0 - w$ for each fixed value of ℓ, where T_ℓ is the number of updates at this value of ℓ.

- $\widetilde{O}\left(\frac{(n+T_\lambda)\log\log W}{\varepsilon^2}\right)$ time for queries to weight estimates for each fixed λ, where T_λ is the number of updates at this value of λ.

ℓ takes on $\widetilde{O}(\log W)$ different values and λ takes on $O\left(\frac{\log W}{\varepsilon}\right)$ different values, and $\sum_\lambda T_\lambda = \sum_\ell T_\ell = T$, so the total time is $\widetilde{O}\left(\frac{(n\log W + T)\log\log W}{\varepsilon^3}\right)$, as desired. □

5.2. Approximating max weight independent set

Now we complete the proof of Theorem 2.3.

Proof. (Proof of Theorem 2.3) Recall that we have a pure covering LP. By Lemma 4.2, it suffices to implement (5.A) in $\widetilde{O}(1/\varepsilon^2)$ amortized time. As remarked above, this reduces to maintain an $(1\pm\varepsilon)$-multiplicative approximation of the deepest point in the weighted range space $(\mathbb{R}^2, R)$, with the weight of a range $r \in R$ being w_r/c_r for nonnegative vector $w : R \to \mathbb{R}_{\geq 0}$ initialized to $\mathbb{1}$ and monotonically decreasing to $n^{-\widetilde{O}(1/\varepsilon)}$, and then straight to 0. By Lemma 5.2, for $T = \widetilde{O}(n/\varepsilon^2)$ and $W = n^{\widetilde{O}(1)}$, such a point can be maintained in $\widetilde{O}\left(\frac{n}{\varepsilon^4}\right)$ total randomized time with high probability, as desired. □

6. Set Multicover via KC Inequalities

In this section we consider the Set Multicover problem. The natural LP relaxation for an instance (P, R, d, c) is the following.

$$\text{minimize} \sum_{r\in R} c_r x_r \text{ over } x : R \to [0,1]$$
$$\text{s.t.} \sum_{r\ni p} x_r \geq d_p \text{ for all } p \in P.$$

An interesting example is when R is a set of disks in the plane. For this case the integrality gap of the preceding LP is known to be $O(1)$ [26, 23]. The LP has both covering and packing constraints and Theorem 2.2 implies that one can compute a $(1\pm\varepsilon)$-approximation in $\widetilde{O}(m+n)$ time if (P, R) admits nearly-linear emptiness oracle with deletions. However, we only obtain a bicriteria approximation. That is, the fractional solution either satisfies the covering constraints exactly or the packing constraints exactly but may not satisfy both. Rounding such a fractional solution can violate the covering constraints by a $(1-\varepsilon)$-factor or we may need to use two copies of each range. We give a slightly refined statement below. Let $d_{\max} = \max_{p\in P} d_p$ be the maximum demand.

Lemma 6.1. *Let x be a $(1\pm\varepsilon)$-approximate solution to the LP for Set Multicover for a range space (P, R). Suppose there is an $\alpha(m,n)$ approximation for instances of Set Multicover over a range space (P, R) via the LP relaxation. Let x be a $(1\pm\varepsilon)$-approximate solution to the LP. Then one can obtain an integer solution of cost $(2+\varepsilon)\alpha(m,n)\,\mathrm{OPT}$ that covers each point $p \in P$ to an extent of $\left\lfloor (1-\varepsilon)d_p + \frac{1}{2}\right\rfloor$. In particular if $\varepsilon < \frac{1}{2d_{\max}}$ then the fractional solution can be rounded to satisfy all the constraints exactly.*

Proof. We will assume that $x \in [0,1]^n$ and hence satisfies the packing constraints exactly, and that it satisfies the covering constraints approximately, that is, for each p $\sum_{r\ni p} x_r \geq (1-\varepsilon)d_p$. Also $\sum_{r\in R} c_r x_r \leq (1+\varepsilon)\,\mathrm{OPT}$. We now consider a new fractional solution x' where $x'_r = \min\{1, 2x_r\}$ for each $r \in R$. It is easy to see that $\sum_{r\ni p} x'_r \geq \left\lfloor \frac{1}{2} + \sum_{r\ni p} x_r \right\rfloor$. Rounding x' gives us the desired solution. □

Thus, we can obtain an algorithm with a running time $\widetilde{O}(\mathrm{poly}(d_{\max})(m+n))$ that yields a $2(1+\varepsilon)\alpha(m,n)$ approximation. If $d_{\max}$ is small (say $\widetilde{O}(1)$) then the running time is reasonable and one would assume that in most applications $d_{\max}$ is likely to be small. It is also easy to obtain a $(1+\varepsilon)\alpha(m,n)(1+\log d_{\max})$-approximation in $\widetilde{O}(m+n)$ time via the bicriteria approximation; iteratively use it in $O(\log d_{\max})$ rounds where in each round we satisfy a constant factor of the residual demand of each point. Now we discuss the case when $d_{\max}$ is large.

LP with Knapsack Cover inequalities: We describe an alternative approach for Set Multicover. The weakness of the bicriteria fractional solution is even more pronounced in the setting of general covering integer programs (CIPs) [7, 12, 44]. A standard solution, following the work of Carr *et al* [7], is to use the so-called Knapsack Cover (KC) inequalities to strengthen the LP. KC inequalities are typically used when the inequalities have large numbers. However, as shown by Quanrud [45], KC inequalities are useful even for a $\{0,1\}$ incidence matrix as is the case here — KC inequalities allow one to convert a mixed packing and covering LP into one with only covering constraints at the expense of adding an exponential number of constraints. The advantage of the KC LP is that even an approximate solution is sufficient for rounding.

We describe the derived LP via KC inequalities for the Set Multicover problem. For $p \in P$ and $S \subseteq R$ we let $R_p = \{r \in R \mid p \in r\}$ denote the set of ranges in R that contain p. Suppose $x \in \{0,1\}^R$ is an integer feasible solution for the LP. Then, for any $p \in P$ and $S \subseteq R_p$ the inequality $\sum_{r \in R_p \setminus S} x_r \geq d_p - |S|$ holds; the quantity $d_p - |S|$ is the residual demand of p assuming that the ranges in S are already included in a partial solution. For ease of notation we let $\operatorname{resd}(p,S)$ denote the quantity $d_p - |S|$ where $S \subseteq R_p$. Based on the preceding observation we can write an LP relaxation for Set Multicover as a *pure covering* LP below.

$$\begin{aligned} \text{minimize } & \sum_{r \in R} c_r x_r \text{ over } x \in \mathbb{R}^R_{\geq 0} \\ \text{s.t. } & \sum_{r \in R_p \setminus S} x_r \geq \operatorname{resd}(p,S) \text{ for all } p \in P,\ S \subset R_p \end{aligned}$$

We note that the preceding LP and the original LP are equivalent since the original LP has a $\{0,1\}$ incidence matrix. However approximate solutions to first LP are weaker than approximate solutions to the second LP. At this point we do not know how to exploit the underlying geometry to solve the LP with KC inequalities in nearly linear time. However, Chekuri and Quanrud [44] showed that for an explicitly specified CIP, the LP with KC inequalities can be solved in $O(N/\varepsilon^3 + (m+n)/\varepsilon^5)$ time where N is the number of nonzeroes in the matrix. In our setting N corresponds to the number of nonzeroes in the matrix $I(P,R)$, which in the worst case can be mn. The advantage of the KC LP is captured by the next lemma.

Lemma 6.2. *Let x be a $(1+\varepsilon)$-approximate solution to the LP with KC inequalities. Then x be rounded to an integer solution of cost $(1+O(\varepsilon))\alpha(m,n)\,\mathrm{OPT}$.*

Proof. Given x let $S = \{r \mid x_r \geq 1-\varepsilon\}$. Let $d'_p = d_p - |S \cap R_p|$ be the residual demand of p after picking S. Since x is an approximate solution to the KC LP we have that $\sum_{r \ni p, r \notin S} x_r \geq (1-\varepsilon) d'_p$ for each p. Consider the fractional solution x' where $x'_r = \min\{1, x_r/(1-\varepsilon)\}$ for each range r. We have $x'_r = 1$ for each $r \in S$. Further, for each p we have $\sum_{r \ni p, r \notin S} x'_r \geq d'_p$. Thus x' is a feasible solution to the original LP and its cost is $\frac{1}{1-\varepsilon}\sum_r c_r x_r \leq (1+O(\varepsilon))\,\mathrm{OPT}$. Thus rounding x' gives us the desired integer solution. □

References

[1] Jon Louis Bentley and James B. Saxe. “Decomposable Searching Problems I: Static-to-Dynamic Transformation”. In: *J. Algorithms* 1.4 (1980), pp. 301–358.

[2] Kenneth L. Clarkson and Peter W. Shor. “Application of Random Sampling in Computational Geometry, II”. In: *Discrete & Computational Geometry* 4 (1989), pp. 387–421.

[3] Kenneth L. Clarkson. “Algorithms for Polytope Covering and Approximation”. In: *Algorithms and Data Structures, Third Workshop, WADS ’93, Montréal, Canada, August 11-13, 1993, Proceedings.* 1993, pp. 246–252.

[4] Michael D. Grigoriadis and Leonid G. Khachiyan. “Fast Approximation Schemes for Convex Programs with Many Blocks and Coupling Constraints”. In: *SIAM Journal on Optimization* 4.1 (1994), pp. 86–107.

[5] Hervé Brönnimann and Michael T. Goodrich. “Almost Optimal Set Covers in Finite VC-Dimension”. In: *Discrete & Computational Geometry* 14.4 (1995), pp. 463–479.

[6] Serge A. Plotkin, David B. Shmoys, and Éva Tardos. “Fast Approximation Algorithms for Fractional Packing and Covering Problems”. In: *Math. Oper. Res.* 20.2 (1995), pp. 257–301.

[7] Robert D. Carr, Lisa Fleischer, Vitus J. Leung, and Cynthia A. Phillips. “Strengthening integrality gaps for capacitated network design and covering problems”. In: *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, January 9-11, 2000, San Francisco, CA, USA.* 2000, pp. 106–115.

[8] A. Efrat, M. J. Katz, F. Nielsen, and M. Sharir. “Dynamic data structures for fat objects and their applications”. In: *Computational Geometry: Theory and Algorithms* 15 (2000), pp. 215–227.

[9] Philip M. Long. “Using the Pseudo-Dimension to Analyze Approximation Algorithms for Integer Programming”. In: *Algorithms and Data Structures, 7th International Workshop, WADS 2001, Providence, RI, USA, August 8-10, 2001, Proceedings.* Ed. by Frank K. H. A. Dehne, Jörg-Rüdiger Sack, and Roberto Tamassia. Vol. 2125. Lecture Notes in Computer Science. Springer, 2001, pp. 26–37.

[10] Neal E. Young. "Sequential and Parallel Algorithms for Mixed Packing and Covering". In: *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*. IEEE Computer Society, 2001, pp. 538–546.

[11] Alon Efrat. "The Complexity of the Union of (alpha, beta)-Covered Objects". In: *SIAM J. Comput.* 34.4 (2005), pp. 775–787.

[12] Stavros G. Kolliopoulos and Neal E. Young. "Approximation algorithms for covering/packing integer programs". In: *J. Comput. Syst. Sci.* 71.4 (2005), pp. 495–505.

[13] Kenneth L Clarkson and Kasturi Varadarajan. "Improved approximation algorithms for geometric set cover". In: *Discrete & Computational Geometry* 37.1 (2007), pp. 43–58.

[14] P. K. Agarwal, J. Pach, and M. Sharir. "State of the Union–of Geometric Objects". In: *Surveys in Discrete and Computational Geometry Twenty Years Later*. Ed. by J. E. Goodman, J. Pach, and R. Pollack. Vol. 453. Contemporary Mathematics. AMS, 2008, pp. 9–48. URL: https://users.cs.duke.edu/~pankaj/publications/surveys/union.pdf.

[15] Boris Aronov and Sariel Har-Peled. "On Approximating the Depth and Related Problems". In: *SIAM J. Comput.* 38.3 (2008), pp. 899–921.

[16] Timothy M. Chan. "A dynamic data structure for 3-D convex hulls and 2-D nearest neighbor queries". In: *J. ACM* 57.3 (2010), 16:1–16:15.

[17] Nabil H. Mustafa and Saurabh Ray. "Improved Results on Geometric Hitting Set Problems". In: *Discrete & Computational Geometry* 44.4 (2010), pp. 883–895.

[18] Kasturi R. Varadarajan. "Weighted geometric set cover via quasi-uniform sampling". In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. Ed. by Leonard J. Schulman. ACM, 2010, pp. 641–648.

[19] Sariel Har-Peled. *Geometric approximation algorithms*. American Mathematical Society, 2011.

[20] Hayim Shaul. "Range Searching: Emptiness, Reporting and Approximate Counting". PhD thesis. Tel-Aviv University, 2011.

[21] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.

[22] Pankaj K. Agarwal, Esther Ezra, and Micha Sharir. "Near-Linear Approximation Algorithms for Geometric Hitting Sets". In: *Algorithmica* 63.1-2 (2012), pp. 1–25.

[23] Nikhil Bansal and Kirk Pruhs. "Weighted geometric set multi-cover via quasi-uniform sampling". In: *European Symposium on Algorithms*. Springer. 2012, pp. 145–156.

[24] Timothy M. Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. "Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling". In: *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*. Ed. by Yuval Rabani. SIAM, 2012, pp. 1576–1585.

[25] Timothy M. Chan and Sariel Har-Peled. "Approximation Algorithms for Maximum Independent Set of Pseudo-Disks". In: *Discrete & Computational Geometry* 48.2 (2012), pp. 373–392.

[26] Chandra Chekuri, Kenneth L. Clarkson, and Sariel Har-Peled. "On the set multicover problem in geometric settings". In: *ACM Trans. Algorithms* 9.1 (2012), 9:1–9:17.

[27] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. "Submodular function maximization via the multilinear relaxation and contention resolution schemes". In: *SIAM Journal on Computing* 43.6 (2014), pp. 1831–1879.

[28] Vincent Cohen-Addad and Claire Mathieu. "The unreasonable success of local search: Geometric optimization". In: *arXiv preprint arXiv:1410.0553* (2014).

[29] Christos Koufogiannakis and Neal E. Young. "A Nearly Linear-Time PTAS for Explicit Fractional Packing and Covering Linear Programs". In: *Algorithmica* 70.4 (2014), pp. 648–674.

[30] Neal E. Young. "Nearly Linear-Time Approximation Schemes for Mixed Packing/Covering and Facility-Location Linear Programs". In: *CoRR* abs/1407.3015 (2014).

[31] Z. Allen-Zhu and L. Orecchia. "Nearly-Linear Time Positive LP Solver with Faster Convergence Rate". In: *Proc. 47th Annu. ACM Sympos. Theory Computing (STOC)*. 2015, pp. 229–236.

[32] Chandra Chekuri, T. S. Jayram, and Jan Vondrák. "On Multiplicative Weight Updates for Concave and Submodular Function Maximization". In: *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*. Ed. by Tim Roughgarden. ACM, 2015, pp. 201–210.

[33] Nabil H Mustafa, Rajiv Raman, and Saurabh Ray. "Quasi-polynomial time approximation scheme for weighted geometric set cover on pseudodisks and halfspaces". In: *SIAM Journal on Computing* 44.6 (2015), pp. 1650–1669.

[34] Di Wang, Satish Rao, and Michael W. Mahoney. "Unified Acceleration Method for Packing and Covering Problems via Diameter Reduction". In: *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*. Ed. by Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi. Vol. 55. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, 50:1–50:13.

[35] Chandra Chekuri and Kent Quanrud. "Approximating the Held-Karp Bound for Metric TSP in Nearly-Linear Time". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*. 2017, pp. 789–800.

[36] Chandra Chekuri and Kent Quanrud. "Near-Linear Time Approximation Schemes for some Implicit Fractional Packing Problems". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*. Ed. by Philip N. Klein. SIAM, 2017, pp. 801–820.

[37] Alina Ene, Sariel Har-Peled, and Benjamin Raichel. "Geometric Packing under Nonuniform Constraints". In: *SIAM J. Comput.* 46.6 (2017), pp. 1745–1784.

[38] Haim Kaplan, Wolfgang Mulzer, Liam Roditty, Paul Seiferth, and Micha Sharir. "Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 2495–2504.

[39] Boris Aronov, Anirudh Donakonda, Esther Ezra, and Rom Pinchasi. "On pseudo-disk hypergraphs". In: *arXiv preprint arXiv:1802.08799* (2018).

[40] Timothy M Chan, Thomas C van Dijk, Krzysztof Fleszar, Joachim Spoerhase, and Alexander Wolff. "Stabbing Rectangles by Line Segments-How Decomposition Reduces the Shallow-Cell Complexity". In: *arXiv preprint arXiv:1806.02851* (2018).

[41] Chandra Chekuri and Kent Quanrud. "Randomized MWU for Postive LPs". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, Louisiana, USA, January 7–10, 2018*. SIAM, 2018.

[42] Anna Adamaszek, Sariel Har-Peled, and Andreas Wiese. "Approximation Schemes for Independent Set and Sparse Subsets of Polygons". In: *J. Assoc. Comput. Mach.* 66.4 (June 2019), 29:1–29:40. ISSN: 0004-5411.

[43] Pankaj K. Agarwal and Jiangwei Pan. "Near-Linear Algorithms for Geometric Hitting Sets and Set Covers". In: *Discrete & Computational Geometry* (2019). Preliminary version appeared in Proc. of SoCG, 2014.

[44] Chandra Chekuri and Kent Quanrud. "On approximating (sparse) covering integer programs". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 1596–1615.

[45] Kent Quanrud. "Fast and Deterministic Approximations for k-Cut". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*. 2019, 23:1–23:20.