WEIGHTED GEOMETRIC SET MULTI-COVER VIA QUASI-UNIFORM SAMPLING *

Nikhil Bansal† *and Kirk Pruhs*‡

Abstract. We give a randomized polynomial-time algorithm with approximation ratio $O(\log \phi(n))$ for weighted set multi-cover instances with a shallow cell complexity of at most $f(z,k) = z\phi(z)k^{O(1)}$. Up to constant factors, this matches a recent result of Chan, Grant, Könemann and Sharpe [6] for the set cover case, i.e. when all the covering requirements are 1. One consequence of this is an $O(1)$-approximation for geometric weighted set multi-cover problems when the geometric objects have linear union complexity; for example when the objects are disks, unit cubes or halfspaces in $\mathbb{R}^3$. Another consequence is that the real difficulty of many natural capacitated set covering problems lies with solving the associated priority cover problem only, and not with the associated multi-cover problem.

1 Introduction

In the *weighted set multi-cover problem* we are given a set $\mathcal{P}$ of n points and a collection $\mathcal{S}$ of m subsets of $\mathcal{P}$. Each element $p \in \mathcal{P}$ has a positive integer demand d_p, and each set $s \in \mathcal{S}$ has a positive weight w_s. A subset X of $\mathcal{S}$ is a feasible multi-cover if each $p \in \mathcal{P}$ lies in (or equivalently is covered by) at least d_p distinct sets in X, and the goal is to find a minimum weight feasible multi-cover. The case when all the demands are unit (i.e. $d_p = 1$ for all p) is known as the weighted set cover problem and has been studied extensively.

It is well known that the natural greedy algorithm is a $1 + \ln n$-approximation algorithm for the weighted set multi-cover problem, and unless NP has slightly super-polynomial-time algorithms, no polynomial-time algorithm can achieve a better approximation ratio (up to lower order terms), even for unit demands and weights [9]. Thus, we focus here on special classes of instances where the underlying structure of the set system allows for an improved approximation. Such classes commonly arise in natural combinatorial optimization problems. For example, some network design problems can be cast as covering a collection of cuts using edges. Another class of such instances arise in geometry, where the sets are geometric objects such as disks, rectangles or fat triangles and the elements are points in $\mathbb{R}^d$. This paper is motivated by the following meta-question:

If a particular set system (or class of set systems) admits a good approximation

*NB was supported in part by a NWO Vidi grant 639.022.211 and an ERC consolidator grant 617951. KP was supported in part by NSF grants CCF-0830558, CCF-1115575, CNS-1253218, CCF-1421508, and an IBM Faculty Award.

†*Eindhoven University of Technology*, n.bansal@tue.nl

‡*University of Pittsburgh*, kirk@cs.pitt.edu

algorithm for the set cover problem, then does it also admit a good approximation algorithm for the multi-cover case?

In addition to its direct relevance, multi-cover problems arise naturally in many practical settings, another important theoretical motivation to study this question arises from the result in [5] that a capacitated set cover problem can be reduced to a priority set cover problem, and a multi-cover problem on similar set systems. Thus, in many cases, better bounds for multi-cover problems will directly improve upon known bounds for capacitated problems.

While the set cover problem has been extensively studied for many classes of set systems, the corresponding multi-cover case has received less attention. At first glance, it seems that there should be a simple general method for extending a set cover result to the corresponding multi-cover case. After all, their natural integer linear programming formulations differ only in the right hand side of the constraints, i.e. $Ax \geq \mathbf{1}$ v.s. $Ax \geq d$. However, this seems unlikely in general. For example, consider the classic survivable network design problem (SNDP): Given a graph $G = (V, E)$ and demands $r(u, v)$ for pairs $u, v \in V$, find a minimum cost subset F of edges such that $G' = (V, F)$ has $r(u, v)$ edge disjoint paths for each pair u, v. SNDP can be viewed as a multi-cover problem where each cut S has covering requirement $d(S) = \max_{u \in S, v \notin S} r(u, v)$. Note that this problem seems much harder than the corresponding set cover case, $r(u, v) \in \{0, 1\}$, which is the Steiner network problem. A 2-approximation algorithm for Steiner network was known long before a 2-approximation was found for SNDP [10]. In fact, obtaining an $O(1)$-approximation for SNDP directly from the Steiner Network result would be a significant breakthrough. Further, extending common techniques for obtaining approximation algorithms for geometric set cover problems to the corresponding multi-cover problems poses several additional challenges (see [7] for a discussion). The only generic connection between set cover and multi-cover that we are aware of is the following (most likely folklore) result [3]: if a set system has an linear programming based α-approximation for set cover, then it has an $O(\min(\log d_{\max}, \log \log n)\alpha)$-approximation for multi-cover, where $d_{\max}$ is the maximum coverage requirement.

In this paper, we study the weighted set multi-cover problem on geometric set systems, and slight generalizations thereof. We extend the best known approximation results for the corresponding weighted set cover problem to hold for weighted multi-cover. Thus, we give a partial answer to the meta-question above: that for common geometric set systems, weighted multi-cover is not harder to approximate than set cover.

1.1 Previous Work

The goal in geometric set cover problems is to improve the $\ln n$ set cover bound by exploiting the underlying geometric structure. This is an active area of research and various different techniques have been developed. However, until recently most of these techniques applied only to the unweighted case. A key idea is the connection between set covers and ϵ-nets [4], which implies that proving better bounds on sizes of ϵ-nets for various geometric systems (an active research area in discrete geometry) directly gives improved guarantees for the unweighted set-cover problem. In another direction, [8] related the guarantee for

unweighted set-cover to the complexity of the geometric objects. Here the complexity of a geometric shape refers to the size needed to describe it (which is roughly the number of vertices, edges, faces etc.). More precisely, [8] showed that if the complexity of the vertical decomposition of the complement of the union of any k sets is $O(kh(k))$, then there is an $O(h(n))$-approximation for unweighted set cover. This was subsequently improved to $O(\log(h(n)))$ [12, 2], and these results were further extended to the (unweighted) multi-cover case in certain "well-behaved" cases [7]. However, none of these techniques work with weights. Roughly speaking, the problem is the following: all these techniques begin by random sampling (say, according to the optimum linear programming solution), followed by an alteration phase where the points that are left uncovered during the sampling phase are now covered. Their techniques are able to show that not many extra sets are needed in the alteration phase. However, they are unable to avoid the possibility that some sets may have a much higher chance of being picked than others, which is problematic if weights are present.

The first breakthrough on weighted geometric cover problems was made by Varadarajan [11], who showed that for geometric set systems with union complexity[1] $O(kh(k))$, there is an efficient randomized algorithm with approximation ratio $O(\exp(O(\log^* n))\log h(n))$. Further, he showed that if the function $h(n)$ is mildly increasing(in particular, it suffices to have $h(n) = \omega(\log^{(j)} n)$ for some constant j), then the approximation ratio improves to $O(\log h(n))$. The key idea behind this result was a new sampling approach, called quasi-uniform sampling, which gives a uniform bound on the factor by which the probability that a set is sampled exceeds the value for that set in the optimal linear programming solution. Recently, Chan et al. [6] refined this approach further and removed the mildly increasing requirement on $h(n)$. They give an $O(\log h(n))$-approximation algorithm for geometric set systems with union complexity $O(nh(n))$. In particular, this algorithm guarantees an $O(1)$-approximation if $h(n) = O(1)$, which for example is the case for disks, pseudo-disks, axis-aligned octants, unit cubes, or half-spaces in three-dimensional space.

Instead of using union complexity, [6] presented their results using the notion of *shallow cell complexity*.

Definition 1. Let $f(z, k)$ be a function that is non-decreasing in both k and z. A set system $\mathcal{S}$ of m sets has shallow cell complexity $f(z, k)$ if for all $1 \leq z \leq m$ and for any collection X of z sets from $\mathcal{S}$, the number of distinct regions covered by k or fewer sets in X is at most $f(z, k)$. Here a region refers to an equivalence class of points that are covered by precisely the same sets in X.

Chan et al. [6] gave an $O(\log \phi(n))$-approximation algorithm for weighted set cover instances with shallow cell complexity $z\phi(z)k^{O(1)}$ (here we assume without loss of generality that $\phi(z) \geq 2$). We remark that Varadarajan [11] also works with shallow cell complexity, without using this terminology directly. In particular, he uses the fact that geometric sets in R^d with union complexity $n\phi(n)$ have shallow cell complexity $O(z\phi(z)k^{d-1})$.

[1] Unlike in [8, 7], the union complexity here means the complexity of the shape of the union of sets, and not of the vertical decomposition of the complement of the union.

1.2 Result and Consequences

Our main result is:

Theorem 2. *There is a randomized polynomial-time algorithm that computes a $c' \log(\phi(n))$-approximation to the minimum weight multi-cover problem on n points where the underlying set system has shallow cell complexity $f(z,k) = z\phi(z)k^c$, for some constant c. Here c' is a universal constant that only depends on c.*

This matches the guarantee (up to constant factors) in [6] for the set cover case, and thus this extends all the geometric covering results in that paper from set cover to the multi-cover case.

An important consequence of our result is for capacitated covering problems. Here, each set s has a capacity c_s in addition to a weight w_s, and each point p has a demand d_p; a solution is feasible if the aggregate capacity of the selected sets covering a point is at least the point's demand. Perhaps surprising at first sight, capacities make the problem substantially more difficult. For example, even if there is just one point to cover (a trivial problem without capacities), we obtain the NP-Hard Knapsack Cover problem. Recently, [5] gave a general method to deal with capacities, and show that any weighted capacitated covering problem can be reduced to weighted multi-cover problem(s) and the weighted priority cover problem on closely related systems. In a weighted priority cover problem, each set and each point has a priority, and each point must be covered by at least one set of higher priority than itself. In particular, they show that an $O(\alpha+\beta)$-approximation algorithm for a weighted capacitated cover problem follows from an α-approximation algorithm for the underlying weighted multi-cover problem, and a β-approximation algorithm for the underlying weighted priority cover problem.

Our result implies that the real bottleneck for approximating capacitated covering problems in geometric settings is solving the associated priority cover problem. This already improves several results where previously the guarantee for the multi-cover problem was a bottleneck. One application is to Theorem 1.3 in [6], which says that any capacitated covering problem on an underlying network matrix has $O(\log\log n)$-approximation. Applying the improved bound in Theorem 2. together with the result in [6] that network matrices have $O(nk)$ shallow cell complexity, improves the approximation ratio to $O(1)$ for such problems. Another application is to a general scheduling problem introduced in [3]. They show that this problem can be cast as a capacitated covering problem where the sets are axis-parallel rectangles with the bottom side touching the x-axis. Applying Theorem 2 to such sets gives an $O(1)$-approximation for the associated multi-cover problem, improving upon the previous $O(\log\log n)$ bound given in [3]. This improves the approximation for the scheduling problem in [3] from $O(\log\log nP)$ to $O(\log\log P)$, where P is the ratio of the maximum to minimum job size.

1.3 Overview of Algorithm Design and Analysis

As mentioned previously, most previous approaches for geometric covering problems work via random sampling. The starting point is an optimal solution to the linear programming

relaxation of the natural integer programming formulation of the problem. If α is the desired approximation, then first the sets s are sampled with probability $O(\alpha x_s)$, where x_s is the probability that s is selected in the optimal linear programming solution. After this step, a forcing procedure is applied (perhaps recursively) to cover the remaining uncovered points. The key insight in [11] and [6] is to develop a forcing procedure where the probability of picking each set s remains bounded by αx_s. Such a forcing is called a quasi-uniform sampling. As our algorithm also uses this framework, we elaborate on this approach a bit more.

Let us start with explaining the approach, specialized to set cover, in which each demand is 1. Without loss of generality, the values x_s can be assumed to be integer multiples of $1/M$ for some large integer M. The rounding proceeds in rounds. In round i, for each set s we independently with probability (roughly) $1/2$, either double x_s, or set x_s to 0. This ensures that each x_s is an integer multiple of $2^i/M$. This continues for roughly $\log M$ rounds, until all variables are close to 0 or 1. In expectation, the extent to which each point is fractionally covered stays (roughly) the same. However, due to the randomness in the doubling step, if some point is p unlucky and is fractionally covered by significantly less sets than expected, then a greedy forcing procedure is invoked to select some sets that will cover p. To obtain a quasi-uniform sampling, we need that the probability of forcing is sufficiently small and that the forcing procedure doesn't force any particular set with too high probability.

We want to adapt the above approach to multi-cover so that no dependence on the covering requirement $d_{\max}$ is incurred in the approximation ratio. The main reason that this is challenging is that a point p must be covered by d_p *distinct* sets, that is, we cannot pick the same set twice. If a set can be picked multiple times, the set-cover analysis in [6] can be adapted very easily to multi-cover (see [7]).

Our basic idea for overcoming this issue is simple. We first pick all sets s with $x_s \geq 1/Q$ for some large enough constant Q, and update the covering requirements accordingly. As the linear programming solution for the residual problem now satisfies $x_s < 1/Q$ for each s, each point p must be fractionally covered by Qd_p sets in the linear programming solution. Thus, if the rounding proceeded according to expectation, there would be enough distinct sets to form a multicover at the end of the rounding. However, if it so happens that the number of distinct sets fractionally covering a point becomes significantly less than expected, then we will again need a forcing procedure to reestablish that condition. The greedy forcing procedure for multicover is the natural extension to the forcing procedure for set cover. And we will again need to argue that the probability of forcing is sufficiently small. However, for multi-cover the argument that the probability of forcing is sufficiently small is significantly more involved than the analogous argument for set cover. This argument is the main technical extension to the analysis for set cover in [6].

The rounding algorithm is described in Section 2. The algorithm's analysis is described in Section 3.

2 The Algorithm Description

The rounding of the linear programming solution proceeds in rounds. We first describe the initial setup that creates the initial instance from the linear programming solution. We then describe the invariant that we wish to maintain at each round. During each round there is a sampling phase that discards some copies of certain sets, a forcing phase that permanently selects some sets and ensures that the invariant in maintained in the next round, and a cleanup phase. We next describe the sampling and cleanup phases, and the termination condition, which are relatively straightforward. The description of the forcing phase is postponed until Subsection 2.1 because it is more involved.

The Initial Setup: We assume that the given set system has shallow cell complexity $f(z,k) = z\phi(z)k^c$ for some constant c and some nondecreasing function $\phi(z)$. To ensure that $\log(\phi(z))$ is positive, we will assume that $\phi(z) \geq 2$.

The exact integer formulation of the weighted set multicover problem is the following:

$$\min \sum_{s\in\mathcal{S}} w_s x_s \qquad \text{s.t.} \sum_{s:p\in s} x_s \geq d_p, \quad \forall p \in \mathcal{P} \quad \text{and} \quad x_s \in \{0,1\}, \quad \forall s \in \mathcal{S},$$

where x_s indicates whether the set s is selected or not.

The algorithm begins by solving the natural linear programming relaxation where we relax the requirement $x_s \in \{0,1\}$ to $x_s \in [0,1]$. Let x denote some fixed basic optimum solution to this linear program. As there are n non-trivial constraints, at most n variables x_s lie strictly between 0 and 1. Let Q be a large enough constant, whose value will be specified later. For each s with $x_s \geq 1/(2Q)$, select s and decrease the demand d_p by 1 for every point p covered by s. Clearly, the cost of selecting these sets is at most $2Q$ times the linear program cost. Consider the residual instance on the remaining sets and points. Let us redefine d_p to be the residual covering requirement of p. Clearly, the residual solution x_s is still feasible for the residual instance.

Let $M = n$. We create a new instance by making $n_s = \lfloor 2Mx_s \rfloor$ copies of each set s. To distinguish between the original sets and the copies in the new instance, we will use *replicas* to denote the copies of a set s. As the linear program solution is feasible, $\sum_{s:p\in s} x_s \geq d_p$ for each point p, and thus p is covered by at least

$$\sum_{s:p\in s} n_s \geq \sum_{s:p\in s, x_s>0} (2Mx_s - 1) \geq \left(\sum_{s:p\in s} 2Mx_s\right) - n \geq M\,(2d_p - 1) \geq Md_p \qquad (1)$$

replicas in the new instance. The second inequality uses the fact that the linear program support is at most n.

Note that in this new instance, each point p is covered by at least Md_p replicas of sets containing p, and each set has at most $n_s \leq 2Mx_s \leq M/Q$ replicas, as $x_s \leq 1/(2Q)$. So p is covered by replicas corresponding to at least Qd_p distinct sets. Let N denote the number of replicas in the instance. As there are at most n sets in the support and $n_s \leq 2M = 2n$ for each s, we have that $N \leq 2n^2$. As $N = O(n^2)$ and $\phi(z) \leq z^d$ for geometric problems in dimension d, $\log(\phi(N)) = O(\log(\phi(n)))$.

The Invariant: We first state the invariant, in equation (2), then give some intuition, and finally formally define all the terms used in the invariant.

$$\sum_{s:p\in s} \min\left(n_s(i), \frac{k(i)}{b(i)}\right) \geq k(i)d_p(i) \quad \forall p. \tag{2}$$

This says that each point must be covered by at least $k(i)d_p(i)$ replicas, but each set is only allowed to contribute up to $k(i)/b(i)$ replicas. In particular this implies that there are at least $b(i)d_p(i)$ distinct sets that can cover p. Actually our algorithm will also ensure that $n_s(i) \leq k(i)/b(i)$ for each set s, but it is more convenient to state the invariant (2) this way.

The term $n_s(i)$ is the number of replicas of set s at the start of round i. The term $d_p(i)$ is the residual covering requirement for point p at the start of round i. The term $k(i) = M/2^{i-1}$ is the number of replicas we would expect for each set at the start of round i if we retained replicas with probability exactly $1/2$. We define

$$\epsilon(i) = R\sqrt{\frac{\log k(i) + \log \phi(N)}{k(i)}} \tag{3}$$

for some sufficiently large constant R. In a couple of places we will need that R is sufficiently large so that we can argue that some bad events are sufficiently rare. The quantity $\epsilon(i)$ represents the bias in the rounding in round i towards retaining a set. Finally the term

$$b(i) = \frac{Q}{\prod_{j=1}^{i-1}(1+4\epsilon(j))}$$

roughly represents a lower bound on the ratio of the number distinct sets covering a point p to the demand of p. We choose Q large enough so that in the final round r, $b(r) = 2$. This completely determines Q. In Claim 7 we will show that $Q = O(1)$.

Clearly, $n_s(1) = n_s$, $k(1) = M$, $d_p(1) = d_p$, and $b(1) = Q$. Thus by noting that $n_s(1) \leq M/Q = k(1)/b(1)$, we can see that invariant (2) initially holds.

A Generic Round: The algorithm does the following in each round i for which $\epsilon(i) < 1/2$.

- *Sampling Phase:* For each set s, independently retain each replica with probability $1/2 + \epsilon(i)$, and discard the rest. Let $n_s^*(i)$ denote the number of retained replicas for set s.
- *Forcing Phase:* After the sampling phase, if the invariant (2) is violated for a point p, that is if

  $$\sum_{s:p\in s} \min\left(n_s^*(i), \frac{k(i+1)}{b(i+1)}\right) < k(i+1)d_p(i),$$

  then some sets covering p are permanently selected (also referred to as forced) to ensure that invariant (2) holds at the start of next round. Details are given in Subsection 2.1. All the replicas of a forced set s are removed, and the residual covering requirement of points in s is reduced by 1. This will be the only point in the algorithm where the residual covering requirement of a point changes.

- *The Cleanup Phase:* For each s, if $n^*_s(i) > k(i+1)/b(i+1)$, we retain $k(i+1)/b(i+1)$ arbitrary replicas of s and discard the rest. Thus we will also have the invariant that the number of replicas of s in the next round $n_s(i+1) = \min(n^*_s(i), k(i+1)/b(i+1))$.

Termination: When $\epsilon(i)$ becomes greater than or equal to $1/2$ in the final round r, then all the sets that have one or more replicas left are selected.

2.1 The Forcing Phase

We now explain how the sets to be forced are selected in a generic round i. Roughly speaking, all points that violate invariant (2) and have the same residual demand are considered in each sub-round. To determine which sets to pick to cover these points, an ordered list of replicas of sets is constructed carefully. This ordering ensures that property that not too many points that can force a set to be picked.

First we define the pseudo-depth of a point, which is a more convenient notion to work with than residual requirement. Then we show in Claim 4 that the pseudo-depth is an upper bound on the residual covering requirement. We then explain, for each possible pseudo-depth q, how to construct an ordered list L_q of replicas that have the property that they cover at least one point of pseudo-depth q, and that is increasingly ordered by the number of equivalent regions that they cover. We then explain how to use these lists to determine the sets to force. Finally, in Claim 5 we show why this forcing ensures that the invariant (2) holds.

Definition 3. The *pseudo-depth* $q_p(i)$ of point p is $\lfloor \sum_{s:p\in s} n_s(i)/k(i) \rfloor$, i.e. the number of replicas covering p at the start of round i, divided by $k(i)$.

Claim 4. If invariant (2) holds at the start of round i, then for all p, $q_p(i) \geq d_p(i)$.

Proof. Invariant (2) implies that for each p, $\sum_{s:p\in s} n_s(i) \geq k(i)d_p(i)$. Since $d_p(i)$ is an integer, this implies that $q_p(i) \geq d_p(i)$. □

Constructing the list L_q: For $q = 1, \ldots,$ let P_q be the collection of all points with pseudo-depth exactly q. Let C_q denote the collection of all replicas that cover some point in P_q. For a collection of sets, a *region* is an equivalence class of points that are all covered by exactly the same sets. We initialize L_q the empty order, and initialize the collection C'_q of remaining replicas to C_q We construct an ordering L_q of the replicas in C_q by iterating the following steps.

1. Select a replica $r \in C'_q$ that covers the least number of regions.
2. Append the replica r to the end of the list L_q, and remove it from C'_q.

Without loss of generality we can assume that all the replicas corresponding to the same set appear consecutively in L_q (as removing a replica r does not change the regions formed by C_q unless r was the last remaining replica of its set). Thus, we can also view

L_q as an ordering of the sets. Let $L_{p,q}$ denote the ordered sublist of L_q of sets that cover point p. Given a point p and a set $s \in L_{p,q}$ we define the rank $\rho_{p,q}(s)$ of s as the number of distinct sets in $L_{p,q}$ that lie no later than s in the ordering (i.e. there are exactly $\rho_{p,q}(s)-1$ distinct sets before s in $L_{p,q}$).

Using the lists L_q to determine which sets to force: Let s_q be the last set $s \in L_q$ for which there is some point p such that $p \in P_q$ and $p \in s$ for which the following forcing condition holds:

$$\sum_{t \in L_{p,q}: \rho_{p,q}(t) \geq \rho_{p,q}(s)} \min\left(n_t^*(i), \frac{k(i+1)}{b(i+1)}\right) < k(i+1)\left(q - (\rho_{p,q}(s)-1)\right). \tag{4}$$

The forcing condition says that the total number of replicas of sets that contain p from s until the end of the list L_q are insufficient to satisfy the invariant (2) (in the next round) even if we pick all the replicas of the $\rho_{p,q}(s)-1$ sets appearing before s. Note that s_q may not exist. If s_q exists, then for each list $L_{p,q}$ all the sets with rank $\rho_{p,q}(s_q)$ or less are forced.

Claim 5. If the forced sets are selected for the multicover, and residual demands are updated appropriately, then invariant (2) will hold for all points.

Proof. We consider a pseudo-depth q. We need to show that the sets forced by pseudo-depth q cause invariant (2) to hold for points of pseudo-depth q, and does not cause invariant (2) to be violated for other pseudo-depths.

First assume that s_q does not exist. Let p be a point with pseudo-depth q. So in particular the forcing condition does not apply for the set s with $\rho_{p,q}(s) = 1$. Hence

$$\sum_{s: p \in s} \min\left(n_s^*(i), \frac{k(i+1)}{b(i+1)}\right) \geq k(i+1)q \geq k(i+1)d_p(i) \geq k(i+1)d_p(i+1)$$

and thus invariant (2) holds for p at the start of the next round.

Now assume that s_q exists, and again let p be a point with pseudo-depth q. Let s'_q be the set with $\rho_{p,q}(s'_q) = \rho_{p,q}(s_q)+1$ that appears after s_q in the ordering $L_{p,q}$. The set s'_q exists as s_q cannot be the last in $L_{p,q}$. This is because the invariant (2) after phase i ensures that $|L_{p,q}| = \sum_{s \in L_{p,q}} n_s(i) \geq b(i)q \geq 2q \geq q+1$, and it must necessarily be that $\rho_{p,q}(s_q) \leq q$ for (4) to hold. As the forcing condition does not hold for p and s'_q

$$\begin{aligned} \sum_{t \in L_{p,q}: \rho_{p,q}(t) \geq \rho_{p,q}(s'_q)} \min\left(n_t^*(i), \frac{k(i+1)}{b(i+1)}\right) &\geq k(i+1)\left(q - (\rho_{p,q}(s'_q)-1)\right) \\ &= k(i+1)\left(q - \rho_{p,q}(s_q)\right). \end{aligned} \tag{5}$$

As the sets in $L_{p,q}$ of rank $\leq \rho_{p,q}(s_q)$ will be in the multicover, the residual requirement for p reduces from $d_p(i)$ to at most $d_p(i) - \rho_{p,q}(s_q)$ in round $i+1$. So equation (5) ensures that the invariant (2) holds for p at the start of round $i+1$.

Finally we show that the forcing for pseudo-depth q does not cause invariant (2) to be violated for a point p of pseudo-depth not equal to q. If a set s covering p is forced, the

residual demand $d_p(i)$ decreases by 1. So the right hand side of invariant (2) reduces by $k(i+1)$. In contrast, the contribution $\min(n_s^*(i), k(i+1)/b(i+1))$ on the left hand side of invariant (2) reduces by at most $k(i+1)/b(i+1) \leq k(i+1)$. □

3 The Algorithm Analysis

We first show that the algorithm returns a feasible multicover upon termination, and then focus on bounding the cost of the solution.

3.1 Feasibility

The correctness of the algorithm follows directly by the following argument.

Claim 6. The algorithm returns a valid multi-cover upon termination.

Proof. As invariant (2) holds at the last round r, each point p satisfies

$$\sum_{s:p\in s, n_s(r)>0} \frac{k(r)}{b(r)} \geq k(r)d_p(r),$$

and hence is covered by replicas involving at least $b(r)d_p(r)$ distinct sets. As $b(r) = 2 > 1$, and each set s with $n_s(r) > 0$ is picked upon termination, the solution forms a valid multi-cover. □

3.2 Bounding the cost via Quasi-Uniformity

We now prove that the algorithm produces a quasi-uniform sample, that is, every set is sampled with probability at most $O(\log \phi(N))$ times[2] its LP contribution x_s.

First we show that $Q = O(1)$ and thus the algorithm only loses an $O(1)$ by rounding all $x_s \geq 1/Q$ to 1. This calculation is similar to Claim 2 in [6], but we describe it in detail here for completeness.

Claim 7. If Q is chosen so that $b(r) = 2$, then $Q = O(1)$.

Proof. As $b(i) = Q/\prod_{j=1}^{i-1}(1+4\epsilon(j))$, using $1+x \leq \exp(x)$, we obtain that $b(i) \geq Q\exp(-4\sum_j \epsilon(j))$, where the summation is over all the rounds j until $\epsilon(j) \leq 1/2$. Recall that $\epsilon(j) = \Theta\left(\sqrt{\frac{\log k(j)+\log\phi(N)}{k(j)}}\right)$. As $k(j)$ decreases geometrically as $M/2^{j-1}$ it follows that $\epsilon(j)$ also increases geometrically (in particular it varies either as $\sqrt{\log k(j))/k(j)}$ if $k(j) \geq \phi(N)$ or as $\sqrt{\log(\phi(N))/k(j)}$ otherwise) and thus $\sum_{i:\epsilon(i)\leq 1/2} \epsilon(i) = O(1)$. □

[2] As $N = O(n^2)$ and $\phi(n) \leq n^d$ for geometric problem in dimension d, $\log(\phi(N)) = O(\log(\phi(n)))$, and hence we will use them interchangeably.

We now prove that quasi-uniformity. The key result, which is proved in Lemma 10, is that the probability that a set is selected (picked) in round i is at most $\frac{1}{k(i)^2}$. Given Lemma 10, the rest of the analysis follows along the lines in [6], which we give here for completeness. In particular, we have the following claim.

Claim 8. If the forcing probability of a replica in round i is at most $1/k(i)^2$, then the probability that a set s is picked eventually by the solution is $O(x_s \log \phi(N))$.

Proof. If s is never forced until the end of the algorithm, the probability that some fixed replica f of s survives until the end is $\prod_{j=0}^{r-1}(1/2+\epsilon(j))$, which by the calculation in Claim 7 is $O(1)/2^r$, which is $O(\log \phi(N)/M)$ by the choice of r.

On the other hand, if f (or equivalently s) is selected in round i, then this happens with probability at most $1/k(i)^2$ conditioned on having survived until round i. As the probability of surviving until round i is $O(1/2^i)$, the probability that f is ever forced is at most

$$\sum_{i=1}^{r} \frac{O(1)}{2^i} \cdot \frac{1}{k(i)^2} = \sum_{i=1}^{r} O(1) \cdot \frac{2^i}{M^2} = O\left(\frac{1}{M}\right).$$

As there are $x_s M$ replicas of s in the beginning, the probability that s is picked by the algorithm, either at termination or due to forcing is $O(x_s M \cdot \log \phi(N)/M) = O(x_s \log \phi(N))$. □

The rest of the analysis focuses on proving Lemma 10. We begin by stating the standard Chernoff bounds [1] that we will use repeatedly.

Theorem 9. *If $X = X_1 + \ldots + X_n$ where each X_i is a 0-1 random variables and $\mu = E[X]$, then for any $0 < \delta < 1$, $Pr[X \leq (1-\delta)\mu] \leq \exp(-\delta^2\mu/2)$ and $Pr[X \geq (1+\delta)\mu] \leq \exp(-\delta^2\mu/4)$.*

Lemma 10. The probability that a set u is selected in round i is at most $\frac{1}{k(i)^2}$.

Proof. Consider some fixed round i and some set u. For notational convenience, let us denote $k(i)$ by k. The set u is picked when the forcing rule (4) applies for a particular pair (p, s) such $p \in u$ and $p \in s$, and s appears no earlier than u in the ordering given by $L_{p,q}$. By the construction of the list L_q, a replica of set u was placed in the list when there were at most $k(q+1)f(N, k(q+1))/N = \phi(N)(k(q+1))^{c+1}$ points $p \in u$ (recall that points within a region are combinatorially identical, so we can assume there is exactly one point per region). As each point p has pseudo-depth q there are at most $k(q+1)$ sets that can contain it. Thus there are at most $\phi(N)(k(q+1))^{c+2}$ pairs (p, s) that can cause u to be picked.

Let us fix a set s and a point p at pseudo-depth q. Let us define the event $\mathcal{E}_{p,q,s}$ that the pair (p, s) is forced at pseudo-depth q.

We will show that

$$\Pr[\mathcal{E}_{p,q,s}] = O((kq\phi(N))^{-c-4}). \tag{6}$$

This will imply Lemma 10 as follows. By a union bound over all possible values of q, and all possible pairs (p, s) it follows that the probability that set u is picked is

$$\sum_{p,s}\sum_{q\geq 1}\Pr[\mathcal{E}_{p,q,s}]\cdot\phi(N)(k(q+1))^{c+2} = O((kq\phi(N))^{-2}) = O(1/k^2).$$

We now focus on showing (6). For notational simplicity we will drop p and q from the subscripts in $L_{p,q}$ and $\rho_{p,q}(s)$. We will also drop i from $b(i)$ and $\epsilon(i)$.

By the definition of forcing (4), we can write the event $\mathcal{E}_{p,q,s}$, that we denote as $\mathcal{E}$ for notational convenience, as

$$\mathcal{E}: \sum_{t\in L:\rho(t)\geq\rho(s)} \min\left(n_t^*, \frac{k(1+4\epsilon)}{2b}\right) < \frac{k}{2}(q-(\rho(s)-1)).$$

To bound $\mathcal{E}$, it is convenient to define two simpler events $\mathcal{E}_1, \mathcal{E}_2$ and bound their probabilities.

$$\mathcal{E}_1: \sum_{t\in L:\rho(t)\geq\rho(s)} n_t^* \leq (1+\epsilon)\left(\frac{kq}{2} - \frac{k(\rho(s)-1)}{4}\right), \tag{7}$$

$$\mathcal{E}_2: \sum_{t\in L:\rho(t)\geq\rho(s)} \left(\mathbf{1}_{n_t^* > \frac{k(1+4\epsilon)}{2b}}\right)\left(n_t^* - \frac{k(1+4\epsilon)}{2b}\right) \geq \frac{\epsilon kq}{2}. \tag{8}$$

These events are useful for the following reason.

Claim 11. For $\mathcal{E}$ to happen, it must be that (i) $\rho(s) \leq q$, and (ii) at least one of the following events $\mathcal{E}_1$ or $\mathcal{E}_2$ must occur.

Proof. If $\rho(s) \geq q+1$, then the right hand side of $\mathcal{E}$ is at most 0 and $\mathcal{E}$ cannot occur.

For the second part we argue as follows. For any c, d, as $\min(c,d) = c - \mathbf{1}_{c>d}(c-d)$, note that the left hand side of $\mathcal{E}$ is identical to the difference of the left hand sides of $\mathcal{E}_1$ and $\mathcal{E}_2$. Thus if neither $\mathcal{E}_1$ or $\mathcal{E}_2$ occurs, then

$$\begin{aligned}\sum_{t\in L:\rho(t)\geq\rho(s)} \min\left(n_t^*, \frac{k(1+4\epsilon)}{2b}\right) &\geq (1+\epsilon)\left(\frac{kq}{2} - \frac{k(\rho(s)-1)}{4}\right) - \frac{\epsilon kq}{2} &(9)\\ &\geq \left(\frac{kq}{2} - \frac{k(\rho(s)-1)}{2}\right) &(10)\end{aligned}$$

where the last step uses that $\epsilon \leq 1$. This implies that $\mathcal{E}$ does not occur either. □

We now bound $\Pr[\mathcal{E}_1]$ and $\Pr[\mathcal{E}_2]$.

Bounding $\Pr[\mathcal{E}_1]$: Let us denote $X = \sum_{t \in L:\rho(t)>\rho(s)} n_t^*$. Note that as each replica is sampled with probability $1/2+\epsilon$, X is the sum of independent 0-1 random variables with mean

$$\mu = \left(\frac{1}{2}+\epsilon\right) \sum_{t \in L:\rho(t)\geq\rho(s)} n_t.$$

Lemma 12. For any set s,

$$\sum_{t \in L:\rho(t)\geq\rho(s)} n_t \geq \left(kq - k\frac{(\rho(s)-1)}{2}\right),$$

and thus $\mu \geq (\frac{1}{2}+\epsilon)(kq - k\frac{(\rho(s)-1)}{2})$.

Proof. As the invariant (2) holds at the beginning of round i for point p, and the pseudo-depth of p is q, by definition 3 this implies that $\sum_{t\in L} n_t \geq kq$. As $n_t \leq k/b \leq k/2$ for each set t and exactly $\rho(s)-1$ sets appear before s in L, it follows that

$$\sum_{t \in L:\rho(t)\geq\rho(s)} n_t \geq \sum_{t\in L} n_t - (k/2)(\rho(s)-1)$$

which implies the result. □

Using the notation above, observe that $\mathcal{E}_1$ is simply the event $X \leq (1+\epsilon)\frac{\mu}{1+2\epsilon} = \left(1-\frac{\epsilon}{1+2\epsilon}\right)\mu$. As $\epsilon \leq 1/2$ and $\mu = (1/2+\epsilon)\mathrm{E}[X] \leq \mathrm{E}[X]$, we have that

$$\Pr[\mathcal{E}_1] \leq \Pr\left[X \leq \left(1-\frac{\epsilon}{3}\right)\mathrm{E}[X]\right].$$

By Theorem 9, this is at most

$$\Pr\left[X \leq \left(1-\frac{\epsilon}{3}\right)\mathrm{E}[X]\right] \leq \exp\left(-\frac{\epsilon^2\mathrm{E}[X]}{18}\right) \leq \exp\left(-\frac{\epsilon^2\mu}{18}\right) \leq \exp\left(-\frac{\epsilon^2 kq}{72}\right). \tag{11}$$

The last inequality holds for the following reason. By Claim 11, $(\rho(s)-1) \leq q$, and by Lemma 12, $\mu \geq (kq - k\frac{(\rho(s)-1)}{2})$ which together imply that $\mu \geq kq/4$.

By the definition of ϵ in (3), we have that $\epsilon^2 kq = \Omega((\log k + \log\phi(N))q)$, and the constant in $\Omega(\cdot)$ can be made arbitrary large by increasing the constant R in (3). Thus by (11), we have that

$$\begin{aligned} \Pr[\mathcal{E}_1] &\leq \exp(-\Omega((\log k + \log\phi(N))q)) \\ &\leq 1/(kq\phi(N))^{-c-5}. \end{aligned} \tag{12}$$

Bounding $\Pr[\mathcal{E}_2]$: Call set a t bad if $n_t^* \geq k(1+4\epsilon)/(2b)$, and let

$$Y_t := \mathbf{1}_{n_t^* > k(1+4\epsilon)/2b}$$

be the 0-1 random variable that indicates t is bad. As $n_t^* \leq n_t$ and $\sum_{t\in L} n_t \leq k(q+1)$ (by the definition of pseudo-depth), there can be at most $2b(q+1)/(1+4\epsilon) \leq 4bq$ candidate bad sets, and thus

$$\sum_{t\in L} Y_t \leq 4bq.$$

Lemma 13. For any set t, $\Pr[Y_t = 1] \leq \min(\epsilon/16, (k\phi(N))^{-c-5})$.

Proof. As $n_t \leq k/b$, and n_t^* is obtained by sampling each replica of t with probability $1/2+\epsilon = (1+2\epsilon)/2$, $\mathrm{E}[n_t^*] \leq (1+2\epsilon)k/2b$. By theorem 9

$$\begin{aligned}\Pr[Y_t = 1] &= \Pr\left[n_t^* \geq \frac{k(1+4\epsilon)}{2b}\right] \\ &\leq \Pr\left[n_t^* \geq \frac{1+4\epsilon}{1+\epsilon}\mathrm{E}[n_t^*]\right] \\ &\leq \Pr\left[n_t^* \geq (1+\epsilon)\mathrm{E}[n_t^*]\right], \end{aligned} \tag{13}$$

where the last step follows as $(1+4\epsilon)/(1+2\epsilon) \leq (1+\epsilon)$ for $\epsilon \leq 1/2$.

As $n_t^* \leq n_t$, it must necessarily hold that $n_t \geq k(1+4\epsilon)/(2b)$ for t to be bad. So we can assume for the sets we care about that $\mathrm{E}[n_t^*] = (1/2+\epsilon)n_t \geq k/8b$. Thus (13) gives us that

$$\Pr[Y_t = 1] = \exp(-\epsilon^2 k/32b) \leq (k\phi(N))^{-2c-12}.$$

The last inequality follows as $b = O(1)$ by Claim 7, and as $\epsilon^2 k = \Theta(\log k + \log\phi(N))$ by (3) and as we can pick the constant in $\Theta(\cdot)$ large enough.

Finally by (3), we have that $\epsilon = \Omega(1/\sqrt{k})$ and thus $k^{-2c-12} \leq \epsilon/16$. □

Lemma 14. $\Pr[\mathcal{E}_2] \leq \Pr[\sum_t Y_t \geq \epsilon bq/2]$.

Proof. We will show that if $\mathcal{E}_2$ holds, then at least $\epsilon bq/2$ sets must be bad. As $n_t^* \leq n_t \leq k/b$, if $\ell < \epsilon bq/2$ sets are bad, then the left hand side of (8) can be at most $\ell(k/b)$, and thus $\mathcal{E}_2$ cannot hold. □

As Y_t is a 0-1 random variable with probability at most $\epsilon/16$ and as there are at most $4bq$ relevant variables (that have non-zero probability of being 1), we have $\mathrm{E}[\sum_t Y_t] \leq \epsilon bq/4$. Thus, by theorem 9 (with $\delta = 1$),

$$\Pr\left[\sum_t Y_t \geq \frac{\epsilon bq}{2}\right] \leq \exp\left(-\frac{\epsilon^2 b^2 q^2}{64}\right). \tag{14}$$

Let us consider two cases. If $b^2 q \geq k$, then the right hand side of (14) is at most $\exp(-\epsilon^2 kq/64)$ which by the choice of ϵ in (3) is at most $(kq\phi(n))^{-c-5}$.

Otherwise, if $b^2q \leq k$, then by Lemma 13 and taking union bound over all the $4bq$ sets,

$$\begin{aligned}\Pr[\sum_t Y_t \geq 1] &\leq \sum_t \Pr[Y_t = 1] \leq 4bq \cdot (k\phi(N))^{-2c-12} \\ &\leq 4 \cdot (k\phi(N))^{-2c-11} \leq 4(kq\phi(N))^{-c-5}.\end{aligned}$$

Here we use that $4bq \leq 4b^2q \leq 4k$, and the last inequality uses that $q \leq k$.

Together this gives

$$\Pr[\mathcal{E}_2] \leq O(kq\phi(N)^{-c-5}). \tag{15}$$

By (12),(15) and Claim 11, it follows that (6) holds which implies Lemma 10 as desired. □

References

[1] N. Alon and J. Spencer. *The Probabilistic Method*. John Wiley, 2008.

[2] Boris Aronov, Esther Ezra, and Micha Sharir. Small-size epsilon-nets for axis-parallel rectangles and boxes. In *ACM Symposium on the Theory of Computing*, pages 639–648, 2009.

[3] Nikhil Bansal and Kirk Pruhs. The geometry of scheduling. In *IEEE Symposium on Foundations of Computer Science*, pages 407–414, 2010.

[4] Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[5] Deeparnab Chakrabarty, Elyot Grant, and Jochen Könemann. On column-restricted and priority covering integer programs. In *Conference on Integer Programming and Combinatorial Optimization*, pages 355–368, 2010.

[6] Timothy M. Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1576–1585, 2012.

[7] Chandra Chekuri, Kenneth L. Clarkson, and Sariel Har-Peled. On the set multi-cover problem in geometric settings. In *Symposium on Computational Geometry*, pages 341–350, 2009.

[8] Kenneth L. Clarkson and Kasturi R. Varadarajan. Improved approximation algorithms for geometric set cover. *Discrete & Computational Geometry*, 37(1):43–58, 2007.

[9] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, July 1998.

[10] Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[11] Kasturi Varadarajan. Weighted geometric set cover via quasi-uniform sampling. In *ACM Symposium on the Theory of Computing*, pages 641–648, 2010.

[12] Kasturi R. Varadarajan. Epsilon nets and union complexity. In *Symposium on Computational Geometry*, pages 11–16, 2009.