Capacitated Covering Problems in Geometric Spaces

Sayan Bandyapadhyay
Department of Computer Science, University of Iowa
Iowa City, USA
sayan-bandyapadhyay@uiowa.edu

Santanu Bhowmick
Department of Computer Science, University of Iowa
Iowa City, USA
santanu-bhowmick@uiowa.edu

Tanmay Inamdar
Department of Computer Science, University of Iowa
Iowa City, USA
tanmay-inamdar@uiowa.edu

Kasturi Varadarajan
Department of Computer Science, University of Iowa
Iowa City, USA
kasturi-varadarajan@uiowa.edu

Abstract

In this article, we consider the following capacitated covering problem. We are given a set P of n points and a set $\mathcal{B}$ of balls from some metric space, and a positive integer U that represents the *capacity* of each of the balls in $\mathcal{B}$. We would like to compute a subset $\mathcal{B}' \subseteq \mathcal{B}$ of balls and assign each point in P to some ball in $\mathcal{B}'$ that contains it, such that the number of points assigned to any ball is at most U. The objective function that we would like to minimize is the cardinality of $\mathcal{B}'$.

We consider this problem in arbitrary metric spaces as well as Euclidean spaces of constant dimension. In the metric setting, even the uncapacitated version of the problem is hard to approximate to within a logarithmic factor. In the Euclidean setting, the best known approximation guarantee in dimensions 3 and higher is logarithmic in the number of points. Thus we focus on obtaining "bi-criteria" approximations. In particular, we are allowed to expand the balls in our solution by some factor, but optimal solutions do not have that flexibility. Our main result is that allowing constant factor expansion of the input balls suffices to obtain constant approximations for this problem. In fact, in the Euclidean setting, only $(1 + \epsilon)$ factor expansion is sufficient for any $\epsilon > 0$, with the approximation factor being a polynomial in $1/\epsilon$. We obtain these results using a unified scheme for rounding the natural LP relaxation; this scheme may be useful for other capacitated covering problems. We also complement these bi-criteria approximations by obtaining hardness of approximation results that shed light on our understanding of these problems.

2012 ACM Subject Classification Theory of computation → Packing and covering problems, Theory of computation → Rounding techniques, Theory of computation → Computational geometry, Mathematics of computing → Approximation algorithms

Keywords and phrases Capacitated covering, Geometric set cover, LP rounding, Bi-criteria approximation

Digital Object Identifier 10.4230/LIPIcs.SoCG.2018.7

Related Version A full version of the paper is available at `https://arxiv.org/abs/1707.05170`

Funding This material is based upon work supported by the National Science Foundation under Grants CCF-1318996 and CCF-1615845

1 Introduction

In this paper, we consider the following capacitated covering problem. We are given a set P of n points and a set $\mathcal{B}$ of balls from some metric space, and a positive integer U that represents the *capacity* of each of the balls in $\mathcal{B}$. We would like to compute a subset $\mathcal{B}' \subseteq \mathcal{B}$ of balls and assign each point in P to some ball in $\mathcal{B}'$ that contains it, such that the number of points assigned to any ball is at most U. The objective function that we would like to minimize is the cardinality of $\mathcal{B}'$. We call this the *Metric Capacitated Covering* (MCC) problem.

An important special case of this problem arises when $U = \infty$, and we refer to this as *Metric Uncapacitated Covering* (MUC). The MUC requires us to cover the points in P using a minimum number of balls from $\mathcal{B}$, and we can therefore solve it using the efficient greedy algorithm for set cover and obtain an approximation guarantee of $O(\log n)$. The approximation factor of $O(\log n)$ for Set Cover cannot be improved unless $P = NP$ [14]. The same is true for the MUC, as demonstrated by the following reduction from Set Cover. We construct a graph with a vertex corresponding to each set and element, and an edge of length 1 between a set and an element if the element is contained in the set. In the metric induced by this graph, we create an MUC instance: we include a ball of radius 1 at each set vertex, and let the points that need to be covered be the element vertices. It is easy to see that any solution for this instance of MUC directly gives a solution for the input instance of the general Set Cover, implying that for MUC, it is not possible to get any approximation guarantee better than the $O(\log n)$ bound for Set Cover.

The MUC in fixed dimensional Euclidean spaces has been extensively studied. One interesting variant is when the allowed set $\mathcal{B}$ of balls consists of all *unit* balls. Hochbaum and Maass [19] gave a polynomial time approximation scheme (PTAS) for this variant using a grid shifting strategy. When $\mathcal{B}$ is an arbitrary finite set of balls, the problem seems to be much harder. An $O(1)$ approximation algorithm in the 2-dimensional Euclidean plane was given by Brönnimann and Goodrich [9]. More recently, a PTAS was obtained by Mustafa and Ray [26]. In dimensions 3 and higher, the best known approximation guarantee is still $O(\log n)$. Motivated by this, Har-Peled and Lee [18] gave a PTAS for a bi-criteria version where the algorithm is allowed to expand the input balls by a $(1+\epsilon)$ factor. Covering with geometric objects other than balls has also been extensively studied; see [22, 12, 4, 27, 10, 17] for a sample.

The MCC is a special case of the *Capacitated Set Cover* (CSC) problem. In the latter problem, we are given a set system $(X, \mathcal{F})$ with $n = |X|$ elements and $m = |\mathcal{F}|$ subsets of X. For each set $\mathcal{F}_i \in \mathcal{F}$, we are also given an integer U_i, which is referred to as its *capacity*. We are required to find a minimum size subset $\mathcal{F}' \subseteq \mathcal{F}$ and assign each element in X to a set in $\mathcal{F}'$ containing it, such that for each set $\mathcal{F}_i$, the number of points assigned to $\mathcal{F}_i$ is at most U_i. The MCC is obtained as a special case of CSC by setting $X = P$, $\mathcal{F} = \mathcal{B}$, and $U_i = U\ \forall i$. Set Cover is a special case of CSC where the capacity of each set is ∞.

Applications of Set Cover include placement of wireless sensors or antennas to serve clients, VLSI design, and image processing [7, 19]. It is natural to consider capacity constraints that appear in many applications, for instance, an upper bound on the number of clients that can be served by an antenna. Such constraints lead to the natural formulation of CSC. For the CSC problem, Wolsey [28] used a greedy algorithm to give an $O(\log n)$ approximation. For the special case of vertex cover (where each element in X belongs to exactly two sets in $\mathcal{F}$), Chuzhoy and Naor [11] presented an algorithm with approximation ratio 3, which was subsequently improved to 2 by Gandhi et al [15]. The generalization where each element belongs to at most a bounded number f of sets has been studied in a sequence of works, culminating in [20, 29]. Berman et al [7] have considered the "soft" capacitated version of the CSC problem that allows making multiple copies of input sets. Another closely related problem to the CSC problem is the so-called *Replica Placement* problem. For the graphs with treewidth bounded by t, an $O(t)$ approximation algorithm for this problem is presented in [1]. Finally, PTASs for the Capacitated Dominating Set, and Capacitated Vertex Cover problems on the planar graphs are presented in [6], under the assumption that the demands and capacities of the vertices are upper bounded by a constant.

Compared to MUC, relatively fewer geometric versions of the MCC problem have been studied in the literature. We refer to the version of MCC where the underlying metric is Euclidean as the *Euclidean Capacitated Covering* (ECC) problem. The dimension of the Euclidean space is assumed to be a constant. One such version arises when $\mathcal{B}$ comprises of all possible unit balls. This problem appeared in the Sloan Digital Sky Survey project [25]. Building on the shifting strategy of Hochbaum and Maass [19], Ghasemi and Razzazi [16] obtain a PTAS for this problem. When the set $\mathcal{B}$ of balls is arbitrary, the best known approximation guarantee is $O(\log n)$, even in the plane.

Given this state of affairs for MCC and ECC, we focus our efforts on finding a bi-criteria approximation. Specifically, we allow the balls in our solution to expand by at most a constant factor λ, without changing their capacity constraints (but optimal solution does not expand). We formalize this as follows. An (α, β)-approximation for a version of MCC, is a solution in which the balls may be expanded by a factor of β (i.e. for any ball B_i, and any point $p_j \in P$ that is assigned to B_i, $d(c_i, p_j) \leq \beta \cdot r_i$), and its cost is at most α times that of an optimal solution (which does not expand the balls). From the reduction of Set Cover to MUC described above, we can see that it is NP-hard to get an $(f(n), \lambda)$-approximation for any $\lambda < 3$ and $f(n) = o(\log n)$. This follows from the observation that in the constructed instance of MUC, the distance between a set vertex and a vertex corresponding to an element not in that set is at least 3. We note that it is a common practice in the wireless network setting to expand the radii of antennas at the planning stage to improve the quality of service. For example, Bose et al [8] propose a scheme for replacing omni-directional antennas by directional antennas that expands the antennas by a constant factor.

Related work. Capacitated version of facility location and clustering type problems have been well-studied over the years. One such clustering problem is the capacitated k-center. In the version of this problem with uniform capacities, we are given a set P of points in a metric space, along with an integer capacity U. A feasible solution to this problem is a choice of k centers to open, together with an assignment of each point in P to an open center, such that no center is assigned more than U points. The objective is to minimize the maximum distance of a point to its assigned center. $O(1)$ approximations are known for this problem [5, 21]; the version with non-uniform capacities is addressed in [2, 13]. Notice that the decision version of the uniform capacitated k-center with the radius parameter r is the same as the decision version of a special case of MCC, where the set $\mathcal{B}$ consists of balls of radius r centered at each point of the capacitated k-center instance. The capacity of each ball is the same as the uniform capacity U of the points. We want to find whether there is a subset of $\mathcal{B}$ consisting of k balls that can serve all the points without violating the capacity constraint. For capacitated versions of other related optimization problems such as metric facility location, k-median etc, see [3, 23, 24] for recent advances.

1.1 Our results and contributions.

In this article, we make significant progress on both the MCC and ECC problems.

- We present an $(O(1), 6.47)$-approximation for the MCC problem. Thus, if we are allowed to expand the input balls by a constant factor, we can obtain a solution that uses at most $O(1)$ times the number of balls used by the optimal solution. As noted above, if we are not allowed to expand by a factor of at least 3, we are faced with a hardness of approximation of $\Omega(\log n)$.
- We present an $(O(\epsilon^{-4d} \log(1/\epsilon)), 1+\epsilon)$-approximation for the ECC problem in $\mathbb{R}^d$. Thus, assuming we are allowed to expand the input balls by an arbitrarily small constant factor, we can obtain a solution with at most a corresponding constant times the number of balls used by the optimal solution. Without expansion, the best known approximation guarantee for $d \geq 3$ is $O(\log n)$, even without capacity constraints.

Both results are obtained via a unified scheme for rounding the natural LP relaxation for the problem. This scheme, which is instantiated in different ways to obtain the two results, may be of independent interest for obtaining similar results for related capacitated covering problems. Though the LP rounding technique is a standard tool that has been used in the literature of the capacitated problems, our actual rounding scheme is different from the existing ones. In fact, the standard rounding scheme for facility location, for example the one in [23], is not useful for our problems, as there a point can be assigned to any facility. But in our case, each point must be assigned to a ball that contains it (modulo constant factor expansion). This hard constraint makes the covering problems more complicated to deal with.

When the input balls have the same radius, it is easier to obtain the above guarantees for the MCC and and the ECC using known results or techniques. For the MCC, this (in fact, even a $(1, O(1))$-approximation) follows from the results for capacitated k-center [5, 21, 13, 2]. This is because of the connection between Capacitated k-center and MCC as pointed out before. The novelty in our work lies in handling the challenging scenario where the input balls have widely different radii. For geometric optimization problems, inputs with objects at multiple scales are often more difficult to handle than inputs with objects at the same scale.

As a byproduct of the rounding schemes we develop, the bicriteria approximations can be extended to a more general capacity model. In this model, the capacities of the balls are not necessarily the same. In particular, suppose ball B_i has capacity U_i and radius r_i. Then for any two balls $B_i, B_j \in \mathcal{B}$, our model assumes that the following holds: $r_i > r_j \implies U_i \geq U_j$. We refer to this capacity model as the *monotonic* capacity model. We refer to the generalizations of the MCC and the ECC problems with the monotonic capacity model as the *Metric Monotonic Capacitated Covering* (MMCC) problem and the *Euclidean Monotonic Capacitated Covering* (EMCC) problem, respectively. We note that the monotonicity assumption on the capacities is reasonable in many applications such as wireless networks – it might be economical to invest in capacity of an antenna to serve more clients, if it covers more area.

Hardness. We complement our algorithmic results with some hardness of approximation results that give a better understanding of the problems we consider. Firstly, we show that for any constant $c > 1$, there exists a constant $\epsilon_c > 0$ such that it is NP-hard to obtain a $(1+\epsilon_c, c)$-approximation for the MCC problem, even when the capacity of all balls is 3. This shows that it is not possible to obtain a $(1, c)$ approximation even for an arbitrarily large constant c. In the hardness construction, not all the balls in the hard instance have the same radius. This should be contrasted with the case where the radii of all balls are equal – in this case one can use the results from capacitated k-center (such as [2, 13]), to obtain a $(1, O(1))$-approximation.

It is natural to wonder if our algorithmic results can be extended to weighted versions of the problems. We derive hardness results that indicate that this is not possible. In particular, we show that for any constant $c \geq 1$, there exists a constant $c' > 0$, such that it is NP-hard to obtain a $(c' \log n, c)$-approximation for the weighted version of MMCC with a very simple weight function (a constant power of original radius).

We describe the natural LP relaxation for the MMCC problem in Section 2. We describe a unified rounding scheme in Section 3, and apply it in two different ways to obtain the algorithmic guarantees for MMCC and EMCC. We refer the reader to the full version of the paper for the results that cannot be accommodated here due to space constraints.

2 LP relaxation for MMCC

Recall that the input for the MMCC consists of a set P of points and a set $\mathcal{B}$ of balls in some metric space, along with an integer capacity $U_i > 0$ for ball $B_i \in \mathcal{B}$. We assume that for any two input balls $B_i, B_j \in \mathcal{B}$, it holds that $r_i > r_j \implies U_i \geq U_j$. The goal is to compute a minimum cardinality subset $\mathcal{B}' \subseteq \mathcal{B}$ for which each point in P can be assigned to a ball $\mathcal{B}'$ containing it in such a way that no more than U_i points are assigned to ball B_i. Let $d(p, q)$ denote the distance between two points p and q in the metric space. Let $B(c, r)$ denote the ball of radius r centered at point c. We let c_i and r_i denote the center and radius of ball $B_i \in \mathcal{B}$; thus, $B_i = B(c_i, r_i)$.

First we consider an integer programming formulation of MMCC. For each set $B_i \in \mathcal{B}$, let $y_i = 1$ if the ball B_i is selected in the solution, and 0 otherwise. Similarly, for each point $p_j \in X$ and each ball $B_i \in \mathcal{B}$, let the variable $x_{ij} = 1$ if p_j is assigned to B_i, and $x_{ij} = 0$ otherwise. We relax these integrality constraints, and state the corresponding linear program as follows:

$$\text{minimize} \quad \sum_{B_i \in \mathcal{B}} y_i \qquad \text{(MMCC-LP)}$$

$$\text{s.t.} \quad x_{ij} \leq y_i \qquad \forall p_j \in P,\ \forall B_i \in \mathcal{B} \qquad (1)$$

$$\sum_{p_j \in P} x_{ij} \leq y_i \cdot U_i \qquad \forall B_i \in \mathcal{B} \qquad (2)$$

$$\sum_{B_i \in \mathcal{B}} x_{ij} = 1 \qquad \forall p_j \in P \qquad (3)$$

$$x_{ij} = 0 \qquad \forall p_j \in P,\ \forall B_i \in \mathcal{B} \text{ such that } p_j \notin B_i \qquad (4)$$

$$x_{ij} \geq 0 \qquad \forall p_j \in P,\ \forall B_i \in \mathcal{B} \qquad (5)$$

$$0 \leq y_i \leq 1 \qquad \forall B_i \in \mathcal{B} \qquad (6)$$

Subsequently, we will refer to an assignment (x, y) that is feasible or infeasible with respect to Constraints 1-6 as just a *solution*. The *cost* of the LP solution $\sigma = (x, y)$ (feasible or otherwise), denoted by $\text{cost}(\sigma)$, is defined as $\sum_{B_i \in \mathcal{B}} y_i$.

3 The algorithmic framework

In this section, we describe our framework for extracting an integral solution from a fractional solution to the above LP. The framework consists of two major steps – Preprocessing and the Main Rounding. The Main Rounding step is in turn divided into two smaller steps – Cluster Formation and Selection of Objects. For simplicity of exposition, we first describe the framework with respect to the MMCC problem as an algorithm and analyze the approximation factor achieved by this algorithm for MMCC. Later, we show how one or more steps of this algorithm can be modified to obtain the desired results for the EMCC.

3.1 The algorithm for the MMCC problem

Before we describe the algorithm we introduce some definitions and notation which will heavily be used throughout this section. For point $p_j \in P$ and ball $B_i \in \mathcal{B}$, we refer to x_{ij} as the *flow* from B_i to p_j; if $x_{ij} > 0$, then we say that the ball B_i *serves* the point p_j. Each ball $B_i \in \mathcal{B}$ can be imagined as a source of at most $y_i \cdot U_i$ units of flow, which it distributes to some points in P.

We now define an important operation, called *rerouting of flow*. "Rerouting of flow for a set $P' \subseteq P$ of points from a set of balls $\mathcal{B}'$ to a ball $B_k \notin \mathcal{B}'$" means obtaining a new solution $(\hat{x}, \hat{y})$ from the current solution (x, y) in the following way: (a) For all points $p_j \in P'$, $\hat{x}_{kj} = x_{kj} + \sum_{B_i \in \mathcal{B}'} x_{ij}$; (b) for all points $p_j \in P'$ and balls $B_i \in \mathcal{B}'$, $\hat{x}_{ij} = 0$; (c) the other $\hat{x}_{ij}$ variables are the same as the corresponding x_{ij} variables. The relevant $\hat{y}_i$ variables may also be modified depending on the context where this operation is used.

Let $0 < \alpha \leq \frac{1}{2}$ be a parameter to be fixed later. A ball $B_i \in \mathcal{B}$ is *heavy* if the corresponding $y_i = 1$, and *light*, if $0 < y_i \leq \alpha$. Corresponding to a feasible LP solution (x, y), let $\mathcal{H} = \{B_i \in \mathcal{B} \mid y_i = 1\}$ denote the set of heavy balls, and $\mathcal{L} = \{B_i \in \mathcal{B} \mid 0 < y_i \leq \alpha\}$ denote the set of light balls. We emphasize that the set $\mathcal{L}$ of light and $\mathcal{H}$ of heavy balls are defined w.r.t. an LP solution; however, the reference to the LP solution may be omitted when it is clear from the context.

Now we move on towards the description of the algorithm. The algorithm, given a feasible fractional solution $\sigma = (x, y)$, rounds σ to a solution $\hat{\sigma} = (\hat{x}, \hat{y})$ such that $\hat{y}$ is integral, and the cost of $\hat{\sigma}$ is within a constant factor of the cost of σ. The $\hat{x}$ variables are non-negative but may be fractional. Furthermore, each point receives unit flow from the balls that are chosen (y values are 1), and the amount of flow each chosen ball sends is bounded by its capacity. Notably, no point gets any non-zero amount of flow from a ball that is not chosen (y value is 0). Moreover, for any ball B_i and any $p_j \in P$, if B_i serves p_j, then $d(c_i, p_j)$ is at most a constant times r_i. We expand each ball by a constant factor so that it contains all the points it serves.

We note that in $\hat{\sigma}$ points might receive fractional amount of flow from the chosen balls. However, since the capacity of each ball is integral, we can find, using a textbook argument for integrality of flow, another solution with the same set of chosen balls, such that the new solution satisfies all the properties of $\hat{\sigma}$ and the additional property, that for each point p, there is a single chosen ball that sends one unit of flow to p [11]. Thus, choosing an optimal LP solution as the input $\sigma = (x, y)$ of the rounding algorithm yields a constant approximation for MMCC by expanding each ball by at most a constant factor.

Our LP rounding algorithm consists of two steps. The first step is a preprocessing step where we construct a fractional LP solution $\overline{\sigma} = (\overline{x}, \overline{y})$ from σ, such that each ball in $\overline{\sigma}$ is either heavy or light, and for each point $p_j \in P$, the amount of flow that p_j can potentially receive from the light balls is at most α. The latter property will be heavily exploited in the next step. The second step is the core step of the algorithm where we round $\overline{\sigma}$ to the desired integral solution.

We note that throughout the algorithm, for any intermediate LP solution that we consider, we maintain the following two invariants: (i) Each ball B_i sends at most U_i units of flow to the points, and (ii) Each point receives exactly one unit of flow from the balls. With respect to a solution $\sigma = (x, y)$, we define the *available capacity* of a ball $B_i \in \mathcal{B}$, denoted $\text{AvCap}(B_i)$, to be $U_i - \sum_{p_j \in P} x_{ij}$. We now describe the preprocessing step.

3.1.1 The preprocessing step

▶ **Lemma 1.** *Given a feasible LP solution $\sigma = (x, y)$, and a parameter $0 < \alpha \leq \frac{1}{2}$, there exists a polynomial time algorithm to obtain another LP solution $\overline{\sigma} = (\overline{x}, \overline{y})$ that satisfies Constraints 1-6 except 4 of MMCC-LP. Additionally, $\overline{\sigma}$ satisfies the following properties.*

1. *Any ball $B_i \in \mathcal{B}$ with non-zero $\overline{y_i}$ is either heavy ($\overline{y_i} = 1$) or light ($0 < \overline{y_i} \leq \alpha$).*
2. *For each point $p_j \in P$, we have that*

$$\sum_{B_i \in \mathcal{L}: \overline{x}_{ij} > 0} \overline{y_i} \leq \alpha, \tag{7}$$

where $\mathcal{L}$ is the set of light balls with respect to $\overline{\sigma}$.

3. *For any heavy ball B_i, and any point $p_j \in P$ served by B_i, $d(c_i, p_j) \leq 3r_i$.*
4. *For any light ball B_i, and any point $p_j \in P$ served by B_i, $d(c_i, p_j) \leq r_i$.*
5. $\text{cost}(\overline{\sigma}) \leq \frac{1}{\alpha}\text{cost}(\sigma)$.

Proof. The algorithm starts off by initializing $\overline{\sigma}$ to σ. While there is a violation of Inequality 7, we perform the following steps.

1. We pick an arbitrary point $p_j \in P$, for which Inequality 7 is not met. Let $\mathcal{L}_j$ be a subset of light balls serving p_j such that $\alpha < \sum_{B_i \in \mathcal{L}_j} \overline{y}_i \leq 2\alpha$. Note that such a set $\mathcal{L}_j$ always exists because the $\overline{y}_i$ variables corresponding to light balls are at most $\alpha \leq \frac{1}{2}$. Let B_k be a ball with the largest radius from the set $\mathcal{L}_j$. (If there is more than one ball with the largest radius, we consider one having the largest capacity among those. Throughout the paper we follow this convention.) Since $r_k \geq r_m$ for all other balls $B_m \in \mathcal{L}_j$, we have, by the *monotonicity* assumption, that $U_k \geq U_m$.
2. We set $\overline{y}_k \leftarrow \sum_{B_i \in \mathcal{L}_j} \overline{y}_i$, and $\overline{y}_m \leftarrow 0$ for all $B_m \in \mathcal{L}_j \setminus \{B_k\}$. Note that $\overline{y}_k \leq 2\alpha \leq 1$. Let $A = \{p_t \in P \mid \overline{x}_{it} > 0 \text{ for some } B_i \in \mathcal{L}_j \setminus \{B_k\}\}$ be the set of "affected" points. We reroute the flow for all the affected points in A from $\mathcal{L}_j \setminus \{B_k\}$ to the ball B_k. Since $U_k \geq U_m$ for all other balls $B_m \in \mathcal{L}_j$, B_k has enough available capacity to "satisfy" all "affected" points. In $\overline{\sigma}$, all other $\overline{x}_{ij}$ and $\overline{y_i}$ variables remain same as before. (*Note*: Since B_k had the largest radius from the set $\mathcal{L}_j$, all the points in A are within distance $3r_k$ from its center c_k, as seen using the triangle inequality. Also, since $\overline{y}_k > \alpha$, B_k is no longer a light ball.)

Finally, for all balls B_i such that $\overline{y}_i > \alpha$, we set $\overline{y}_i = 1$, making them heavy. Thus $\text{cost}(\overline{\sigma})$ is at most $\frac{1}{\alpha}$ times $\text{cost}(\sigma)$, and $\overline{\sigma}$ satisfies all the conditions stated in the lemma. ◀

Remark. As a byproduct of Lemma 1, we get a simple $(4, 3)$-approximation algorithm for the *soft* capacitated version of our problem that allows making multiple copies of the input balls.

3.1.2 The main rounding step

The main rounding step can be logically divided into two stages. The first stage, *Cluster Formation*, is the crucial stage of the algorithm. Note that there can be many light balls in the preprocessed solution. Including all these balls in the final solution may incur a huge cost. Thus we use a careful strategy based on flow rerouting to select a small number of balls. The idea is to use the capacity of a selected light ball to reroute as much flow as possible from other intersecting balls. This in turn frees up some capacity at those balls. The available capacity of each heavy ball is used, when possible, to reroute *all* the flow from some light ball intersecting it; this light ball is then added to a cluster centered around the heavy ball. Notably, for each cluster, the heavy ball is the only ball in it that actually serves some points, as we have rerouted flow from the other balls in the cluster to the heavy ball. In the second stage, referred to as *Selection of Objects*, we select exactly one ball (in particular, a largest ball) from each cluster as part of the final solution, and reroute the flow from the heavy ball to this ball, and expand it by the required amount. Together, these two stages ensure that we do not end up choosing many light balls.

We now describe the two stages in detail. Recall that any ball in the preprocessed solution is either heavy or light. Also $\mathcal{L}$ denotes the set of light balls and $\mathcal{H}$ the set of heavy balls. Note that any heavy ball B_i may serve a point p_j which is at a distance $3r_i$ from c_i. We expand each heavy ball by a factor of 3 so that B_i can contain all points it serves.

1. **Cluster formation.** In this stage, each light ball, will be added to either a set $\mathcal{O}$ (that will eventually be part of the final solution), or a cluster corresponding to some heavy ball. Till the very end of this stage, the sets of heavy and light balls remain unchanged. The set $\mathcal{O}$ is initialized to $\emptyset$. For each heavy ball B_i, we initialize the cluster of B_i, denoted by cluster(B_i) to $\{B_i\}$. We say a ball is clustered if it is added to a cluster.
   At any point, let Λ denote the set consisting of each light ball that is (a) not in $\mathcal{O}$, and (b) not yet clustered. While the set Λ is non-empty, we perform the following steps.
   a. While there is a heavy ball B_i and a light ball $B_t \in \Lambda$ such that (1) B_t intersects B_i; and (2) AvCap(B_i) is at least the flow $\sum_{p_j \in P} \overline{x}_{tj}$ out of B_t:
      1. For all the points served by B_t, we reroute the flow from B_t to B_i.
      2. We add B_t to cluster(B_i).
   After the execution of this *while* loop, if the set Λ becomes empty, we stop and proceed to the *Selection of Objects* stage. Otherwise, we proceed to the following.
   b. For any ball $B_j \in \Lambda$, let $\mathcal{A}_j$ denote the set of points currently being served by B_j. Also, for $B_j \in \Lambda$, let $k_j = \min\{U_j, |\mathcal{A}_j|\}$, i.e. k_j denotes the minimum of its capacity, and the number of points that it currently serves. We select the ball $B_t \in \Lambda$ with the maximum value of k_j, and add it to the set $\mathcal{O}$.
   c. Since we have added B_t to the set $\mathcal{O}$ that will be among the selected balls, we use the available capacity at B_t to reroute flow to it. This is done based on the following three cases depending on the value of k_t.
      1. $k_t = |\mathcal{A}_t| \le U_t$. In this case, for each point p_l in B_t that gets served by B_t, we reroute the flow of p_l from $\mathcal{B} \setminus \mathcal{O}$ to B_t. Note that after the rerouting, p_l is no longer being served by a ball in Λ. The rerouting increases the available capacity of other balls intersecting B_t. In particular, for each $B_i \in \mathcal{H}$, AvCap(B_i) increases by $\sum_{p_l : B_t \text{ serves } p_l} \overline{x}_{il}$.
      2. $k_t = U_t < |\mathcal{A}_t|$, but $k_t = U_t > 1$. Observe that the flow out of ball B_t is $\sum_{p_j \in \mathcal{A}_t} x_{tj} \le \alpha U_t$; thus AvCap($B_t$) $\ge (1-\alpha)U_t = (1-\alpha)k_t$.
      In this case, we select a point $p_j \in \mathcal{A}_t$ arbitrarily, and reroute the flow of p_j from

$\mathcal{B} \setminus \mathcal{O}$ to B_t. This will increase the available capacity of other balls in $\mathcal{B} \setminus \mathcal{O}$ that were serving p_j. Also note that p_j is no longer being served by a ball in Λ.

We repeat the above flow rerouting process for other points of $\mathcal{A}_t$ until we encounter a point p_l such that rerouting the flow of p_l from $\mathcal{B} \setminus \mathcal{O}$ to B_t violates the capacity of B_t. Thus the flow assignment of p_l remains unchanged. Note that we can reroute the flow of at least $\lfloor (1-\alpha)k_t \rfloor = \lfloor (1-\alpha)U_t \rfloor \geq 1$ points of $\mathcal{A}_t$ in this manner, since $U_t > 1$ and $\alpha \leq 1/2$.

3. $k_t = U_t = 1 < |\mathcal{A}_t|$. Note that B_t has used $\sum_{p_j \in \mathcal{A}_t} x_{tj} \leq \alpha U_t = \alpha$ capacity. In this case, we pick a point $p_j \in \mathcal{A}_t$ arbitrarily, and then perform the following two steps:

(i) Reroute the flow of p_j from Λ to B_t; after this, p_j is no longer being served by a ball in Λ. Note that in this step, we reroute at most α amount of flow. Therefore, at this point we have $\mathrm{AvCap}(B_t) \geq 1 - 2\alpha$. Let f be the amount of flow p_j receives from the balls in $\mathcal{O}$.

(ii) Then we reroute $\min\{\mathrm{AvCap}(B_t), 1-f\}$ amount of flow of p_j from the set $\mathcal{H}$ to B_t.

When the loop terminates, we have that each light ball is either in $\mathcal{O}$ or clustered. We set $\overline{y}_i \leftarrow 1$ for each ball $B_i \in \mathcal{O}$, thus making it heavy. For convenience, we also set $\mathrm{cluster}(B_i) = \{B_i\}$ for each $B_i \in \mathcal{O}$. Note that, throughout the algorithm we ensure that, if a point $p_j \in P$ is currently served by a ball $B_i \in \Lambda$, then the amount of flow p_j receives from any ball $B_{i'} \in \mathcal{O} \cup \Lambda$ is the same as that in the preprocessed solution, i.e., the flow assignment of p_j w.r.t. the balls in $\mathcal{O} \cup \Lambda$ remains unchanged.

2. **Selection of objects.** At the start of this stage, we have a collection of clusters, each centered around a heavy ball, such that the light balls in each cluster intersect the heavy ball. We are going to pick exactly one ball from each cluster and add it to a set $\mathcal{C}$. Let $\mathcal{C} = \emptyset$ initially. For each heavy ball B_i, we consider $\mathrm{cluster}(B_i)$ and perform the following steps.

   a. If $\mathrm{cluster}(B_i)$ consists of only the heavy ball, we add B_i to $\mathcal{C}$.

   b. Otherwise, let B_j be a largest ball in $\mathrm{cluster}(B_i)$. If $B_j = B_i$, then we expand it by a factor of 3. Otherwise, B_j is a light ball intersecting with B_i, in which case we expand it by a factor of 5. In this case, we also reroute the flow from the heavy ball to the selected ball B_j. Note that since we always choose a largest ball in the cluster, its capacity is at least that of the heavy ball, because of the *monotonicity* assumption. We add B_j to $\mathcal{C}$, and we set $\overline{y}_s \leftarrow 0$ for any other ball B_s in the cluster.

   After processing the clusters, we set $\overline{y}_t \leftarrow 1$ for each ball $B_t \in \mathcal{C}$. Finally, we return the current set of heavy balls (i.e., $\mathcal{C}$) as the set of selected balls. Note that the flow out of each such ball is at most its capacity, and each point receives one unit of flow from the (possibly expanded) balls that contain it. As mentioned earlier, this can be converted into an integral flow.

3.1.3 The analysis of the rounding algorithm

Let OPT be the cost of an optimal solution. We establish a bound on the number of balls our algorithm outputs by bounding the size of the set $\mathcal{C}$. Then we conclude by showing that any input ball that is part of our solution expands by at most a constant factor to cover the points it serves.

For notational convenience, we refer to the solution $\overline{\sigma} = (\overline{x}, \overline{y})$ at hand after preprocessing, as $\sigma = (x, y)$. Now we bound the size of the set $\mathcal{O}$ computed during Cluster Formation. The basic idea is that each light ball added to $\mathcal{O}$ creates significant available capacity in the heavy balls. Furthermore, whenever there is enough available capacity, a heavy ball clusters intersecting light balls, thus preventing them from being added to $\mathcal{O}$. The actual argument is more intricate because we need to work with a notion of y-accumulation, a proxy for available capacity. The way the light balls are picked for addition to $\mathcal{O}$ plays a crucial role in the argument.

Let $\mathcal{H}_1$ (resp. $\mathcal{L}_1$) be the set of heavy (resp. light) balls after preprocessing, and I be the total number of iterations in the Cluster Formation stage. Also let L_j be the light ball selected (i.e. added to $\mathcal{O}$) in iteration j for $1 \leq j \leq I$. Now, L_t maximizes k_j amongst all balls from Λ in iteration t (Recall that k_j was defined as the minimum of the number of points being served by L_j, and its capacity). Note that $k_1 \geq k_2 \geq \cdots \geq k_I$. For any $B_i \in \mathcal{H}_1$, denote by $F(L_t, B_i)$, the total amount of flow rerouted in iteration t from B_i to L_t corresponding to the points B_i serves. This is the same as the increase in $\text{AvCap}(B_i)$ when L_t is added to $\mathcal{O}$. Correspondingly, we define $Y(L_t, B_i)$, the "y-credit contributed by L_t to B_i", to be $\frac{F(L_t,B_i)}{k_t}$. Now, the increase in available capacity over all balls in $\mathcal{H}_1$ is $F_t = \sum_{B_i \in \mathcal{H}_1} F(L_t, B_i)$. The approximation guarantee of the algorithm depends crucially on the following simple lemma, which states that in each iteration we make "sufficiently large" amount of flow available for the set of heavy balls.

▶ **Lemma 2.** *Consider a ball $B_t \in \mathcal{O}$ processed in the* Cluster Formation *stage, step c. For $0 < \alpha \leq 3/8$, $F_t \geq \frac{1}{5} k_t$.*

Proof. The algorithm ensures that the flow assignment of each point in $\mathcal{A}_t$ w.r.t. the balls in $\mathcal{O} \cup \Lambda$ is the same as that in the preprocessed solution. Thus by property 2 of Lemma 1, each such point gets at most α amount of flow from the balls in $\mathcal{O} \cup \Lambda$. Now there are three cases corresponding to the three substeps of step c.

1. $k_t = |\mathcal{A}_t| \leq U_t$. For each point in $\mathcal{A}_t$, at most α amount of flow comes from the balls in $\mathcal{O} \cup \Lambda$. So the remainder is rerouted from the balls in $\mathcal{H}_1$ resulting in a contribution of at least $1 - \alpha$ towards F_t. Therefore, we get that $F_t \geq (1-\alpha)k_t \geq \frac{1}{5}k_t$, since $0 < \alpha \leq 3/8$.
2. $1 < k_t = U_t < |\mathcal{A}_t|$. It is possible to reroute the flow of at least $\lfloor (1-\alpha)U_t \rfloor = \lfloor (1-\alpha)k_t \rfloor$ points of $\mathcal{A}_t$ from $\mathcal{B} \setminus \mathcal{O}$ to B_t. Therefore, we get that $F_t \geq (1-\alpha)\lfloor (1-\alpha)k_t \rfloor$. When $k_t > 1$, the previous quantity is at least $\frac{1}{5}k_t$, again by using the fact that $0 < \alpha \leq 3/8$.
3. When $1 = k_t = U_t < |\mathcal{A}_t|$, $F_t \geq (1 - 2\alpha) \geq \frac{1}{5}k_t$, as $0 < \alpha \leq 3/8$. ◀

At any moment in the Cluster Formation stage, for any ball $B_i \in \mathcal{H}_1$, define its y-accumulation as

$$\widetilde{y}(B_i) = \left(\sum_{L_t \in \mathcal{O}} Y(L_t, B_i) \right) - \left(\sum_{B_j \in \mathcal{L} \cap \text{ cluster}(B_i)} y_j \right).$$

The idea is that B_i gets y-credit when a light ball is added to $\mathcal{O}$, and loses y-credit when it adds a light ball to cluster(B_i); thus, $\widetilde{y}(B_i)$, a proxy for the available capacity of B_i, indicates the "remaining" y-credit. The next lemma gives a relation between the y-accumulation of B_i and its available capacity.

▶ **Lemma 3.** *Fix a heavy ball $B_i \in \mathcal{H}_1$, and an integer $1 \leq t \leq I$. Suppose that $L_1, L_2, \cdots, L_t$ have been added to $\mathcal{O}$. Then $AvCap(B_i) \geq \widetilde{y}(B_i) \cdot k_t$.*

Proof. The proof is by induction on t. For this proof, we abbreviate $\text{AvCap}(B_i)$ by A_i. In the first iteration, just after adding L_1, $A_i \geq F(L_1, B_i) = Y(L_1, B_i) \cdot k_1 \geq \widetilde{y}(B_i) \cdot k_1$.

Assume inductively that we have added balls $L_1, \cdots, L_{t-1}$ to the set $\mathcal{O}$, and that just after adding L_{t-1}, the claim is true. That is, if $\widetilde{y}(B_i)$ and A_i are, respectively, the y-accumulation and the available capacity of B_i just after adding L_{t-1}, then $A_i \geq \widetilde{y}(B_i) \cdot k_{t-1}$.

Consider the iteration t. At step (a) of Cluster Formation, B_i uses up some of its available capacity to add 0 or more balls to cluster(B_i), after which at step (b) we add L_t to $\mathcal{O}$. Suppose that at step (a), one or more balls are added to cluster(B_i). Let B_j be the first such ball, and let k and C_1 be the number of points B_j serves and the capacity of B_j, respectively. Then the amount of capacity used by B_j is at most

$$\min\{C_1 \cdot y_j, k \cdot y_j\} = \min\{C_1, k\} \cdot y_j \leq k_{t-1} \cdot y_j$$

where the last inequality follows because of the order in which we add balls to $\mathcal{O}$. Now, after adding B_j to cluster(B_i), the new y-accumulation becomes $\widetilde{y}(B_i)' = \widetilde{y}(B_i) - y_j$. As for the available capacity,

$$A_i' \geq A_i - k_{t-1} \cdot y_j \geq (\widetilde{y}(B_i) \cdot k_{t-1}) - k_{t-1} \cdot y_j \geq (\widetilde{y}(B_i) - y_j) \cdot k_{t-1} = \widetilde{y}(B_i)' \cdot k_{t-1}$$

Therefore, the claim is true after addition of the first ball B_j. Note that B_i may add multiple balls to cluster(B_i), and the preceding argument would work after each such addition.

Now consider the moment when L_t is added to $\mathcal{O}$. Let $\widetilde{y}(B_i)$ denote the y-accumulation just before this. Now, the new y-accumulation of B_i becomes $\widetilde{y}(B_i)' = \widetilde{y}(B_i) + Y(L_t, B_i)$. If $\widetilde{y}(B_i) \leq 0$, then the new available capacity is

$$A_i' \geq F(L_t, B_i) = Y(L_t, B_i) \cdot k_t \geq \widetilde{y}(B_i)' \cdot k_t.$$

If $\widetilde{y}(B_i) > 0$, the new available capacity, using the inductive hypothesis, is

$$A_i' \geq \widetilde{y}(B_i) \cdot k_{t-1} + Y(L_t, B_i) \cdot k_t \geq (\widetilde{y}(B_i) + Y(L_t, B_i)) \cdot k_t = \widetilde{y}(B_i)' \cdot k_t$$

where, in the second inequality we use $k_t \leq k_{t-1}$. ◀

Now, in the next lemma, we show that any ball $B_i \in \mathcal{H}_1$ cannot have "too-much" y-accumulation at any moment during Cluster Formation.

▶ **Lemma 4.** *At any moment in the Cluster Formation stage, for any ball $B_i \in \mathcal{H}_1$, we have that $\widetilde{y}(B_i) \leq 1 + \alpha$.*

Proof. The proof is by contradiction. Let $B_i \in \mathcal{H}_1$ be the first ball that violates the condition. As $\widetilde{y}(B_i)$ increases only due to addition of a light ball to set $\mathcal{O}$, suppose L_t was the ball whose addition to $\mathcal{O}$ resulted in the violation.

Let $\widetilde{y}(B_i)$ and $\widetilde{y}(B_i)' = \widetilde{y}(B_i) + Y(L_t, B_i)$ be the y-accumulations of B_i just before and just after the addition of L_t. Because of the assumption, $\widetilde{y}(B_i) \leq 1 + \alpha$. So the increase in the y-accumulation of B_i must be because $Y(L_t, B_i) > 0$. Thus, L_t intersects B_i. However, $Y(L_t, B_i) \leq 1$ by definition. Therefore, we have $\widetilde{y}(B_i) > \alpha$.

Now, by Lemma 3, just before addition of L_t, AvCap(B_i) $\geq \widetilde{y}(B_i) \cdot k_{t-1} > \alpha \cdot k_{t-1} \geq \alpha \cdot k_t$, as $k_t \leq k_{t-1}$. However, L_t is a light ball, and so the total flow out of L_t is at most αk_t. Therefore, the available capacity of B_i is large enough that we can add L_t to cluster(B_i), instead of to the set $\mathcal{O}$, which is a contradiction. ◀

▶ **Lemma 5.** *At the end of Cluster Formation stage, we have $|\mathcal{O}| \leq 5 \cdot \Big((1 + \alpha) \cdot |\mathcal{H}_1| + \sum_{B_j \in \mathcal{L}_1} y_j\Big)$, where $0 < \alpha \leq 3/8$.*

Proof. At the end of Cluster Formation stage,

$$\begin{aligned}
\sum_{B_i \in \mathcal{H}_1} \widetilde{y}(B_i) &\geq \sum_{\substack{B_i \in \mathcal{H}_1 \\ 1 \leq t \leq I}} Y(L_t, B_i) - \sum_{B_i \in \mathcal{H}_1} \sum_{B_j \in \text{ cluster}(B_i)} y_j \\
&\geq \sum_{1 \leq t \leq I} (F_t/k_t) - \sum_{B_j \in \mathcal{L}_1} y_j \\
&\qquad\qquad (\because F_t = \textstyle\sum_{B_i \in \mathcal{H}_1} F(L_t, B_i) = k_t \cdot \sum_{B_i \in \mathcal{H}_1} Y(L_t, B_i)) \\
&\geq \frac{1}{5} \cdot |\mathcal{O}| - \sum_{B_j \in \mathcal{L}_1} y_j \qquad (8)
\end{aligned}$$

Where we used Observation 2 to get the last inequality.
Now, adding the inequality of Lemma 4 over all $B_i \in \mathcal{H}_1$, we have that $\sum_{B_i \in \mathcal{H}_1} \widetilde{y}(B_i) \leq (1+\alpha) \cdot |\mathcal{H}_1|$. Combining this with (8) yields the desired inequality. ◀

▶ **Lemma 6.** *The cost of the solution returned by the algorithm is at most* 21 *times the cost of an optimal solution.*

Proof. Let $\sigma = (x, y)$ be the preprocessed LP solution. Now, the total number of balls in the solution is $|\mathcal{O}| + |\mathcal{H}_1|$. Using Lemma 5,

$$\begin{aligned}
|\mathcal{O}| + |\mathcal{H}_1| &\leq 5 \cdot \left((1+\alpha) \cdot |\mathcal{H}_1| + \sum_{B_j \in \mathcal{L}_1} y_j \right) + |\mathcal{H}_1| \\
&\leq (6 + 5\alpha) \left(\sum_{B_j \in \mathcal{H}_1} y_j + \sum_{B_j \in \mathcal{L}_1} y_j \right) \\
&\leq (6 + 5\alpha) \cdot \text{cost}(\sigma) \\
&\leq \left(\frac{6 + 5\alpha}{\alpha} \right) \cdot OPT = 21 \cdot OPT \qquad \text{(by setting } \alpha = 3/8)
\end{aligned}$$

◀

▶ **Lemma 7.** *In the algorithm each input ball is expanded by at most a factor of* 9*.*

Proof. Recall that when a light ball becomes heavy in the preprocessing step, it is expanded by a factor of 3. Therefore after the preprocessing step, any heavy ball in a solution may be an expanded or unexpanded ball.

Now, consider the selection of the balls in the second stage. If a cluster consists of only a heavy ball, then it does not expand any further. Since it might be an expanded light ball, the total expansion factor is at most 3.

Otherwise, for a fixed cluster, let r_l and r_h be the radius of the largest light ball and the heavy ball, respectively. If $r_l \geq r_h$, then the overall expansion factor is 5. Otherwise, if $r_l < r_h$, then the heavy ball is chosen, and it is expanded by a factor of at most 3. Now as the heavy ball might already be expanded by a factor of 3 during the preprocessing step, here the overall expansion factor is 9. ◀

If the capacities of all balls are equal, then one can improve the expansion factor to 6.47 by using an alternative procedure to the *Selection of Balls* stage. Lastly, from Lemmas 6 and 7, we get the following theorem.

▶ **Theorem 8.** *There is a polynomial time* $(21, 9)$*-approximation algorithm for the MMCC problem.*

3.2 The algorithm for the EMCC problem

Overview of the algorithm. For the EMCC problem, we can exploit the structure of $\mathbb{R}^d$ to restrict the expansion factor of the balls to at most $(1+\epsilon)$, while paying in terms of the cost of the solution. In the following, we give an overview of how to adapt the stages of the framework for obtaining this result. Note that in each iteration of the preprocessing stage for MMCC, we consider a point p_j and a cluster $\mathcal{L}_j$ of light balls. We select a largest ball from this set and reroute the flow of other balls in $\mathcal{L}_j$ to this ball. However, to ensure that the selected ball contains all the points it serves we need to expand this ball by a factor of 3. For the EMCC problem, for the cluster $\mathcal{L}_j$, we consider the bounding hypercube whose side is at most a constant times the maximum radius of the balls from $\mathcal{L}_j$, and subdivide it into multiple cells. The granularity of the cells is carefully chosen to ensure that (1) Selecting a maximum radius ball among the balls whose centers are lying in that cell, and expanding it by $(1+\epsilon)$ factor is enough to cover all such balls, and (2) The total number of cells is $poly(1/\epsilon)$. From each cell we select a maximum radius ball, expand it by $(1+\epsilon)$ factor, and reroute the flow from the other balls in that cell to it. It follows that the cost of the preprocessed solution goes up by at most a $poly(1/\epsilon)$ factor. The Cluster Formation stage for the EMCC problem is exactly the same as that for the MMCC problem. Note that, in the Selection of Balls stage for MMCC, we select only one ball per cluster and expand it by $O(1)$ factor to cover all the balls in the cluster. But in case of EMCC, we want to restrict the expansion factor of the balls to at most $(1+\epsilon)$. Hence we select $poly(1/\epsilon)$ number of balls per cluster in a way so that the selected balls when expanded by a factor of $(1+\epsilon)$ can cover all the balls in the cluster. The ideas are similar to the ones in the Preprocessing stage, however one needs to be more careful to handle some technicalities that arise. We summarize the result for the EMCC problem in the following theorem.

▶ **Theorem 9.** *There is a polynomial time $(O(\epsilon^{-4d}\log(1/\epsilon)), 1+\epsilon)$-approximation algorithm for the EMCC problem in $\mathbb{R}^d$, for any $\epsilon > 0$.*

References

1 Anshul Aggarwal, Venkatesan T. Chakaravarthy, Neelima Gupta, Yogish Sabharwal, Sachin Sharma, and Sonika Thakral. Replica placement on bounded treewidth graphs. In *Algorithms and Data Structures - 15th International Symposium, WADS 2017, St. John's, NL, Canada, July 31 - August 2, 2017, Proceedings*, pages 13–24, 2017. doi:10.1007/978-3-319-62127-2_2.

2 Hyung-Chan An, Aditya Bhaskara, Chandra Chekuri, Shalmoli Gupta, Vivek Madan, and Ola Svensson. Centrality of trees for capacitated k-center. *Math. Program.*, 154(1-2):29–53, 2015. doi:10.1007/s10107-014-0857-y.

3 Hyung-Chan An, Mohit Singh, and Ola Svensson. Lp-based algorithms for capacitated facility location. In *FOCS*, pages 256–265, 2014.

4 Boris Aronov, Esther Ezra, and Micha Sharir. Small-size ϵ-nets for axis-parallel rectangles and boxes. *SIAM J. Comput.*, 39(7):3248–3282, 2010. doi:10.1137/090762968.

5 Judit Bar-Ilan, Guy Kortsarz, and David Peleg. How to allocate network centers. *J. Algorithms*, 15(3):385–415, 1993. URL: http://dblp.uni-trier.de/db/journals/jal/jal15.html#Bar-IlanKP93.

6 Amariah Becker. Capacitated dominating set on planar graphs. In *Approximation and Online Algorithms - 15th International Workshop, WAOA 2017, Vienna, Austria, September 7-8*, 2017.

7 Piotr Berman, Marek Karpinski, and Andrzej Lingas. Exact and approximation algorithms for geometric and capacitated set cover problems. *Algorithmica*, 64(2):295–310, 2012.

8 Prosenjit Bose, Paz Carmi, Mirela Damian, Robin Y. Flatland, Matthew J. Katz, and Anil Maheshwari. Switching to directional antennas with constant increase in radius and hop distance. *Algorithmica*, 69(2):397–409, 2014.

9 Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995. doi:10.1007/BF02570718.

10 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*, pages 1576–1585, 2012. URL: https://dl.acm.org/citation.cfm?id=2095241.

11 Julia Chuzhoy and Joseph Naor. Covering problems with hard capacities. *SIAM J. Comput.*, 36(2):498–515, 2006.

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

13 Marek Cygan, MohammadTaghi Hajiaghayi, and Samir Khuller. LP rounding for k-centers with non-uniform hard capacities. In *FOCS*, pages 273–282, 2012.

14 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633, 2014. doi:10.1145/2591796.2591884.

15 Rajiv Gandhi, Eran Halperin, Samir Khuller, Guy Kortsarz, and Srinivasan Aravind. An improved approximation algorithm for vertex cover with hard capacities. *J. Comput. Syst. Sci.*, 72(1):16–33, 2006.

16 Taha Ghasemi and Mohammadreza Razzazi. A PTAS for the cardinality constrained covering with unit balls. *Theor. Comput. Sci.*, 527:50–60, 2014.

17 Sathish Govindarajan, Rajiv Raman, Saurabh Ray, and Aniket Basu Roy. Packing and covering with non-piercing regions. In *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, pages 47:1–47:17, 2016. doi:10.4230/LIPIcs.ESA.2016.47.

18 Sariel Har-Peled and Mira Lee. Weighted geometric set cover problems revisited. *JoCG*, 3(1):65–85, 2012.

19 Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32(1):130–136, 1985.

20 Mong-Jen Kao. Iterative partial rounding for vertex cover with hard capacities. In *SODA*, pages 2638–2653, 2017.

21 Samir Khuller and Yoram J. Sussmann. The capacitated K-center problem. *SIAM J. Discrete Math.*, 13(3):403–418, 2000.

22 Nissan Lev-Tov and David Peleg. Polynomial time approximation schemes for base station coverage with minimum total radii. *Computer Networks*, 47(4):489–501, 2005.

23 Retsef Levi, David B. Shmoys, and Chaitanya Swamy. Lp-based approximation algorithms for capacitated facility location. *Math. Program.*, 131(1-2):365–379, 2012. doi:10.1007/s10107-010-0380-8.

24 Shi Li. On uniform capacitated k-median beyond the natural LP relaxation. In *SODA*, pages 696–707, 2015.

25 Robert Lupton, F. Miller Maley, and Neal E. Young. Data collection for the sloan digital sky survey - A network-flow heuristic. *J. Algorithms*, 27(2):339–356, 1998. doi:10.1006/jagm.1997.0922.

26 Nabil H. Mustafa and Saurabh Ray. Improved results on geometric hitting set problems. *Discrete & Computational Geometry*, 44(4):883–895, 2010. doi:10.1007/s00454-010-9285-9.

27 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*, pages 641–648, 2010. doi:10.1145/1806689.1806777.

28 Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

29 Sam Chiu-wai Wong. Tight algorithms for vertex cover with hard capacities on multigraphs and hypergraphs. In *SODA*, pages 2626–2637, 2017.